НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ імені ІГОРЯ СІКОРСЬКОГО»
Факультет Електроніки
(повна назва інституту/факультету)

Кафедра акустичних та мультимедійних електронних систем
(повна назва кафедри)

«До захисту допущено»

В.о. завідувача кафедри
_____ С. А. Найда
(підпис) (ініціали, прізвище)

«01» червня 2020 р.

Дипломна робота

на здобуття ступеня бакалавра

зі спеціальності (спеціалізації) 171 - Електроніка
(код та назва спеціальності)

на тему: «Спектральний аналіз вокальних звуків»

Виконав: студент __4_ курсу, групи ДГ-г61-1
(шифр групи)

Денисенко Олександр Ігорович _____ _____
(прізвище, ім'я, по батькові) (підпис)

Керівник ___професор каф. АМЕС, д.т.н. Продеус А.М. _____
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Консультант _____ _____ _____
(назва розділу) (посада, вчене звання, науковий ступінь, прізвище, ініціали) (підпис)

Рецензент доцент каф. ЕІ, к.т.н. Шуляк О.П. _____ _____
(посада, науковий ступінь, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.
Студент _____
(підпис)

Київ – 2020 року

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»

Факультет Електроніки
(повна назва)

Кафедра Акустичних та мультимедійних електронних систем
(повна назва)

Рівень вищої освіти – перший (бакалаврський)

Спеціальність (спеціалізація) **171 – Електроніка**
(код і назва)

ЗАТВЕРДЖУЮ
В.о. завідувача кафедри
В.о. завідувача кафедри
_____ С. А. Найда
(підпис) (ініціали, прізвище)
«01» червня 2020 р.

ЗАВДАННЯ
на дипломний проект (роботу) студенту
Денисенко Олександру Ігоровичу
(прізвище, ім'я, по батькові)

1. Тема проекту (роботи) «Спектральний аналіз вокальних звуків»

керівник проекту (роботи) Продеус Аркадій Миколайович, д.т.н., професор,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від «25» 05 2020 р. № 1196-с

2. Строк подання студентом роботи 14 червня 2020 р.

3. Вихідні дані до роботи Наукові джерела з питань моделювання, програмного забезпечення, приклади програмних рішень спектрального аналізу стану голосового тракту

4. Зміст (дипломної роботи) пояснювальної записки (перелік завдань, які потрібно розробити)

Дослідити сучасну систему аналізу стану голосового тракту, визначити місце відповідних програмних продуктів, обгрунтувати доцільність обраної теми дослідження, виявити невирішені проблеми, протиріччя, порівняти досягнення вітчизняних та зарубіжних фахівців з проблеми дослідження, розробити прототип програмної частини апаратно-програмного комплексу аналізу стану голосового тракту.

5. Перелік ілюстративного матеріалу (із зазначенням обов'язкових плакатів, презентацій тощо) Презентація із 15 слайдів

6. Консультанти розділів проекту (роботи)*

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7. Дата видачі завдання 1 листопада 2019 р.

Календарний план

№ з/п	Назва етапів виконання дипломного проекту (роботи)	Строк виконання етапів проекту (роботи)	Примітка
1	Збір та вивчення джерел інформації для написання дипломної роботи; складання бібліографії наукових джерел	13 квітня 2020	Виконано
2	Складання плану дипломної роботи	15 квітня 2020	Виконано
3	Написання дипломної роботи	1 червня 2020	Виконано
4	Оформлення дипломної роботи	5 червня 2020	Виконано
5	Здача оформленої роботи на кафедру	14 червня 2020	Виконано

Студент _____ О. І. Денисенко
(підпис) (ініціали, прізвище)

Керівник проекту (роботи) _____ А. М. Продеус
(підпис) (ініціали, прізвище)

* Консультантом не може бути зазначено керівника дипломного проекту (роботи)

РЕФЕРАТ

Предметом роботи є методи аналізу стану голосового тракту.

Об’єктом є процеси розробки програмної частини апаратно-програмного комплексу аналізу вокальних звуків, спрямовані на забезпечення вимог лікарів-фоніатрів.

Метою дипломної роботи є аналіз існуючих програмних рішень спектрального аналізу голосових сигналів та розробка власного на базі дослідженого та вивченого інструментарію розробки подібних систем.

У даній роботі проведено аналіз можливостей програмного середовища MATLAB від MathWorks та обрано методи та алгоритми, що найкращим чином підходять для виконання поставленої задачі. За обраними методами та алгоритмами проведено аналіз раціональності обраних технічних рішень.

На базі проведеного аналізу виконано розробку власного програмного забезпечення, спрямованого на забезпечення вимог лікарів-фоніатрів. Воно має сучасний вигляд та є гнучким у сенсі можливості його майбутнього удосконалення.

Ключові слова: голосовий тракт, спектральний аналіз, дискретне перетворення Фур’є.

ABSTRACT

The subject of the work is the analysis of the state of the vocal tract.

The object is the software part of the hardware-software complex of vocal sound analysis.

The purpose of the thesis is to analyze existing software solutions for spectral analysis of voice signals and to develop their own on the basis of researched and studied tools for the development of such systems.

This paper analyzes the capabilities of the MATLAB software environment from MathWorks and selects the algorithms that are best suited to perform the task. According to the selected algorithms, the analysis of the influence of different evaluation parameters on its result was performed.

On the basis of the conducted analysis the own software aimed at use by phoniatricians is developed. It has a modern look and is flexible in terms of its future improvement.

Keywords: voice tract, spectral analysis, discrete Fourier transform.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧЕНЬ, ТЕРМІНІВ

УЗ – ультразвук

ЧМФ – час максимальної фонації

ДПФ – дискретне перетворення Фур’є

ШПФ – швидке перетворення Фур’є

ВСТУП

Тенденції розвитку людства демонструють те, що з кожним наступним роком люди стикаються з все більшою кількістю захворювань у всіх сферах здоров’я, які потребують вчасного виявлення та діагностування без доставляння дискомфорту пацієнту у вигляді різноманітних діагностичних операцій, так званих інвазивних процедур, що є досить небажаними етапами визначення типу та причини захворювання пацієнта.

Але спершу лікар користується неінвазивними методами, до прикладів яких можна віднести опитування скарг пацієнта, використання лікарем терапевтом пристрою стетоскопу, візуальний огляд зубним лікарем хворих зубів, тощо, а також системи об’єктивної та суб’єктивної оцінки якості слуху, або перші системи аналізу стану голосового тракту.

Також дуже важливою є допомога лікарю у визначенні патології об’єктивними методами, спираючись на показники або параметри, що надаються певним чином автоматизованими системами, що створюються інженерами. Так виникли перші системи рентгенографії, де використовуються рентгенівське випромінювання, згодом ультразвукова діагностика з використанням механічних коливань тканин людини, а вже потім системи магнітно-резонансної томографії, де використовується явище ядерного магнітного резонансу тканин органів людини. Дані системи є досить точними та діючими серед наявних неінвазивних процедур. Проте дієвими вони є не у всіх випадках, причому подібний аналіз є досить коштовним для пацієнта.

Проблема патології голосу властива досить великій групі ризику, до якої входять вчителі, вихователі, артисти та співаки, працівники медіа, журналісти, ведучі, оператори call-центрів, та в тій чи іншій мірі кожен з нас. Саме тому питання розробки та подальшого підтримування системи аналізу вокальних звуків набуло актуальності.

Специфіка розробки подібних систем полягає у тому, що розроблений інженером апаратно-програмний комплекс має бути зручним для використання лікарем і актуальним в сенсі наявних рішень по виявленню певних параметрів тієї

чи іншої патології. Це накладає суттєві складності на роботу інженера, оскільки потребує від нього не лише якостей розробника, але й певним чином дизайнера, що тягне за собою необхідність бачити проблему з точки зору користувача. До таких проблем належать питання вибору апаратного забезпечення (кількість та потужність обраних приладів), що є економічно розумним для максимальної кількості споживачів, але при цьому достатніми для виконання поставленої задачі, а також питання програмного забезпечення (інтерфейсні та обчислювальні рішення).

Саме це питання і не має наразі остаточної відповіді щодо спектрального аналізу стану голосового тракту, що й призвело до постановки задачі по розробці відповідного програмного забезпечення. Із існуючих програмних рішень по допомозі лікарю-фоніатру можна навести лише декілька, з огляду на інтерфейс яких можна сказати, що вони не оптимізовані для роботи будь-якого користувача, окрім самого інженера.

Метою даної роботи є створення програмної частини програмно-апаратного комплексу для аналізу акустичних параметрів голосу в нормі та при паталогіях з метою подальшої діагностики та лікування.

Основною задачею роботи є розробка нового програмного забезпечення з зручним для лікаря-користувача інтерфейсом та усіма можливими обчислювальними методами, що призводить до поліпшення отриманого результату.

Результати роботи можуть бути використані лікарем-фоніатром для полегшення діагностики та подальшого лікування патологій голосу.

РОЗДІЛ 1
МЕТОДИ АНАЛІЗУ СТАНУ ГОЛОСОВОГО ТРАКТУ

У даному розділі наведено аналіз методів діагностики патологій голосу. Розглянуті як технічні рішення, так і алгоритми без застосування технічних засобів, а також огляд існуючих програмних рішень.

1.1 Загальний аналіз методів вирішення задачі

Фоніатрія - галузь оториноларингології, яка вивчає патологію голосу (дисфонія), методи лікування та профілактику голосових розладів, а також способи корекції голосу. Голосові розлади можуть бути проявом захворювання, анатомічних особливостей, а також проявом психологічних факторів. Вирішення деяких проблем може бути тісно пов'язане як з проблемами логопедії, так і з лікуванням супутніх оториноларингологічних захворювань. До таких захворювань належать ларингіт (гострий та хронічний), доброякісні пухлини гортані, функціональна дисфонія, гіпотонічна дисфонія, доброякісні пухлини голосових зв’язок, передвузлові стани голосових зв’язок.

На практиці лікар-фоніатр використовує цілий ряд методик діагностики голосу. З огляду на те, що процес формування звуку реалізується на базі досить складної багатофакторної взаємодії різних систем та механізмів, дослідження стану голосового тракту базується не лише на ньому. Тому, діагностуючи різні голосові розлади, лікарі використовують не лише низку прийомів для всебічного оцінювання стану голосового апарату та якості утвореного ним сигналу, а й, у разі необхідності, алгоритми, що говорять про стан органів та систем, функції яких впливають на голосоутворення.

Вивчення фактичного стану голосового апарату дає можливість виявити різні патологічні порушення, визначити їх розташування, оцінити правильне положення гортані та глотки при співі та мовленні. Оглядаючи пацієнтів з голосовими розладами, часто необхідно вивчити стан органів і систем, тісно пов'язаних з процесом формування голосу. Таким чином, поряд із суто фоніатричними методами

дослідження часто використовуються інструментальні методи дослідження центральної нервової, серцево-судинної, слухової, ендокринної, дихальної, опорно-рухової та інших систем. При необхідності для консультацій та лікування залучаються фахівці різних спеціальностей, що дозволяє покращити діагностику та лікування пацієнтів із голосовими розладами.

Лікар-фоніатр займається відновленням голосу і поверненням усього спектру його можливостей. Результат лікування тим кращий, чим раніше було продіагностовано захворювання гортані та встановлено його причини. У фоніатрії застосовується ціла низка діагностичних методик - загальноклінічні дослідження, оториноларингологічний огляд, збір анамнезу та скарг хворих, спеціалізований фоніатричний огляд з використанням функціональних навантажень, перцептивна оцінка голосу, вимірювання часу максимальної фонації, ларингостробоскопія, мікроларингоскопія, мікроларингостробоскопія, фіброназоларингоскопія, спектральний аналіз голосу, а також визначення “голосового поля”, сонографія, рентгенографія гортані, комп’ютерна та магнітно-резонансна томографія гортані, рентгенокінематографія гортані, електроміографія, хронаксиметрія, глотографія, дослідження амплітуди вібрато і зовнішнього дихання під час співу, електрокімографія, методи дослідження органів і систем, пов’язаних з процесом голосоутворення. Серед алгоритмів дослідження голосової функції та визначення стану голосового апарату найбільш важливими є спеціалізований фоніатричний огляд, перцептивна оцінка голосу, вимірювання часу максимальної фонації, ларингостробоскопія, фіброназоларингоскопія, спектральний аналіз голосу.

При обстеженні пацієнта лікар-фоніатр в першу чергу з’ясовує скарги хворого, визначає час і причину виникнення захворювання з урахуванням умов праці та побуту, анамнестичні дані. Важливо враховувати перенесені раніше захворювання як верхніх дихальних шляхів, так і внутрішніх органів, особливо інфекційні, а також стан серцево-судинної, слухової та нервової систем хворого. Уже під час попередньої розмови з обстежуваним лікар-фоніатр уважно прислуховується до голосу пацієнта, оцінюючи його якості за допомогою власного слухового сприйняття. Лікар має уважно оцінити інтенсивність голосу (тихий, послаблений,

гучний), характер атаки звуків, наявність призвуків, характеристики і ступінь хрипоти, характеристики звуків голосу за висотою, гіпер- або гіпоназальність, порушення темпу та мелодики, постійний чи мінливий характер змін та ін. Такий підхід є найбільш широко вживаним та доступним методом дослідження у фоніатрії. Це зветься методом перцептивної оцінки голосу, тобто оцінки якості голосу на слух дослідника. Потрібно враховувати суб'єктивність слухового сприйняття, беручи до уваги стан слухового органа, диференційність та тренованість слуху, досвід спеціаліста.

Слід зауважити, що голосовий апарат людини є певним акустичним приладом, що призначений для генерації акустичних сигналів та випромінювання їх у оточуюче середовище. В силу цього використання акустичних методів діагностики захворювань і є найбільш зручним та логічним.

Сама по собі перцептивна оцінка голосу може вважатися акустичним методом, оскільки лікар-фоніатр має на слух визначити характеристики голосу та його можливість відтворити звук у тому вигляді, яке буде прийнятне для пацієнта.

Показники, на яких ґрунтується така оцінка, також важко віддаються описові, у зв'язку з чим визначення голосових якостей має нерідко невизначений, метафоричний характер. Особливі складності зустрічаються при оцінці голосу з незначними порушеннями. При застосування перцептивної оцінки слід зважати на те, що вона значною мірою залежить від досвідченості спеціаліста та тренованості його слуху.

Запропоновано декілька шкал оцінки голосу при прослуховуванні, які включають оцінку в балах таких властивостей, як силу голосу, захрипліть, характер дихання та ін. У більшості випадків такі шкали забезпечують певну роздільну здатність від нормального голосу до афонії (відсутність звучного голосу).

Також до акустичних методів можна віднести УЗ діагностику, що не завжди підходить для визначення патологій. Ультразвукове сканування тканин гортані дозволяє визначити деякі об'ємні процеси гортані та прилеглих тканин при умові, що розмір утворень відповідає роздільній здатності приладу (мінімум 1-2мм), а їх розташування близьке до передніх відділів гортані.

Ще одним акустичним методом є часовий аналіз голосу - вимірювання часу максимальної фонації. Клініко-фізіологічне значення цього тесту полягає у тому, що він дає можливість оцінити загальну кількість повітря, що застосовується для утворення звуку, діяльність внутрішніх м’язів гортані та експіраторну напругу дихальних м’язів. Вимірювання часу максимальної фонації проводять наступним чином: просять пацієнта набрати легені повітря і на зручній для нього висоті тону та з комфортною гучністю формувати голосну “а”. За допомогою секундоміра вимірюють час від початку звуку до переходу його в шепіт. Величина ЧМФ може бути використаною для розрахунку коефіцієнта фонації, який відображає витрату повітря при фонації за одиницю часу. Враховуючи те, що при вимірюванні ЧМФ досліджувані використовують майже увесь робочий об'єм легень, коефіцієнт фонації для кожної тональності визначається як відношення життєвої ємності легень до ЧМФ. Чим більший коефіцієнт фонації, тим вищі витрати повітря під час фонації, тим частіші під час голосового навантаження мають бути фонаційно-дихальні цикли (повні і короткі).

Досить поширеним серед дослідників є акустичний спектральний аналіз голосу з використанням комп’ютерної техніки. Спектрографія є об’єктивним методом, за допомогою якого проводиться визначення важливих акустичних характеристик голосу, насамперед частота і амплітуда його основного тону, обертонів, ділянок високої та низької співочих формант, що визначає якісні характеристики голосу.

Як відомо, усі звуки, що використовуються при мовотворенні, можна поділити на дві групи – голосні та приголосні. Перші виникають під дією коливань голосових складок та керуються артикуляційним апаратом. Результуюча голосна фонема видозмінюється в залежності від форми та об’єму ротової порожнини. Досліджено, що кожна голосна фонема набуває на графіку спектрального розподілу потужності характерних посилень, які постійні за висотою і не залежать від частоти основного тону. Дані області підсилення спектральних компонент були названі формантами. Встановлено, що в кожній голосній фонемі є дві основні формантні області. Одна з них пов’язана з резонансом об’єму повітря глотки, а друга – з резонансом об’єму ротової порожнини. При цьому одним з артикуляційних органів є язик, зміна

положення якого в ротовій порожнині та глотці призводить до зміни зазначених вище об’ємів повітря в ротовій порожнині та глотці та утворення тим самим певних формант.

Окрім двох основних формант, в спектрі голосної фонеми може міститися ще 1-3 ділянки підсилення гармонічних складових, які значним чином впливають на формування тембру та розрізнення звуків, але при спектральному аналізі стану голосового тракту використовуються лише деякі.

Далі наведені межі положень формантних областей за різними авторами. На думку І. М. Литвака та Л. А. Варшавського [6], голосні фонеми «у», «о», «а», та «і» є одноформантними, «е» та «и» - двоформантними. М. А. Сапожков та Фант [5] наводять дані із зазначенням трьох формантних областей.

Таблиця 1.1.1. Формантні області за різними авторами

Фонема	а	і	о	у	и	е
Автори	І. М. Литвак та Л. А. Варшавський					
F_1	1100-1400	2800-4200	400-800	200-600	200-600	600-1000
F_2	-	-	-	-	1500-2300	1600-2500
Автори	М. А. Сапожков та Фант					
F_1	700±30	240±30	535±27	300±35	300±22	440±20
F_2	1080±30	2250±38	780±32	625±25	1480±62	1800±48
F_3	2600±65	3200±120	2500±50	2500±55	2330±40	2550±65

На рис.1.1.1 та рис.1.1.2 наведене зображення спектрального аналізу голосового сигналу під час фонації звуку “і”. На першому зображено усереднений спектр фонеми у нормі, а на другому — ця ж фонема у випадку вираженої патології голосового апарату (спастична дисфонія).

О.Г.Павлихин, А.П.Мещеркин [4] зазначають, що досить часто бувають випадки, коли за допомогою звичайних методів дослідження початкові порушення голосу не можуть бути виявлені. Особливо це стосується вокалістів-професіоналів. Наприклад, вокаліст скаржиться на якість голосу, або педагог при прослуховуванні

відзначає недоліки (підхрипування, коливання, неточне інтонування), а при проведенні стандартного фоніатричного обстеження не відзначається якихось значних відхилень. Також залишаються спірними питання визначення типу голосу співака, оскільки анатомічні параметри не завжди відповідають типу голосу. В таких випадках корисним буде застосування спектрального аналізу голосу.

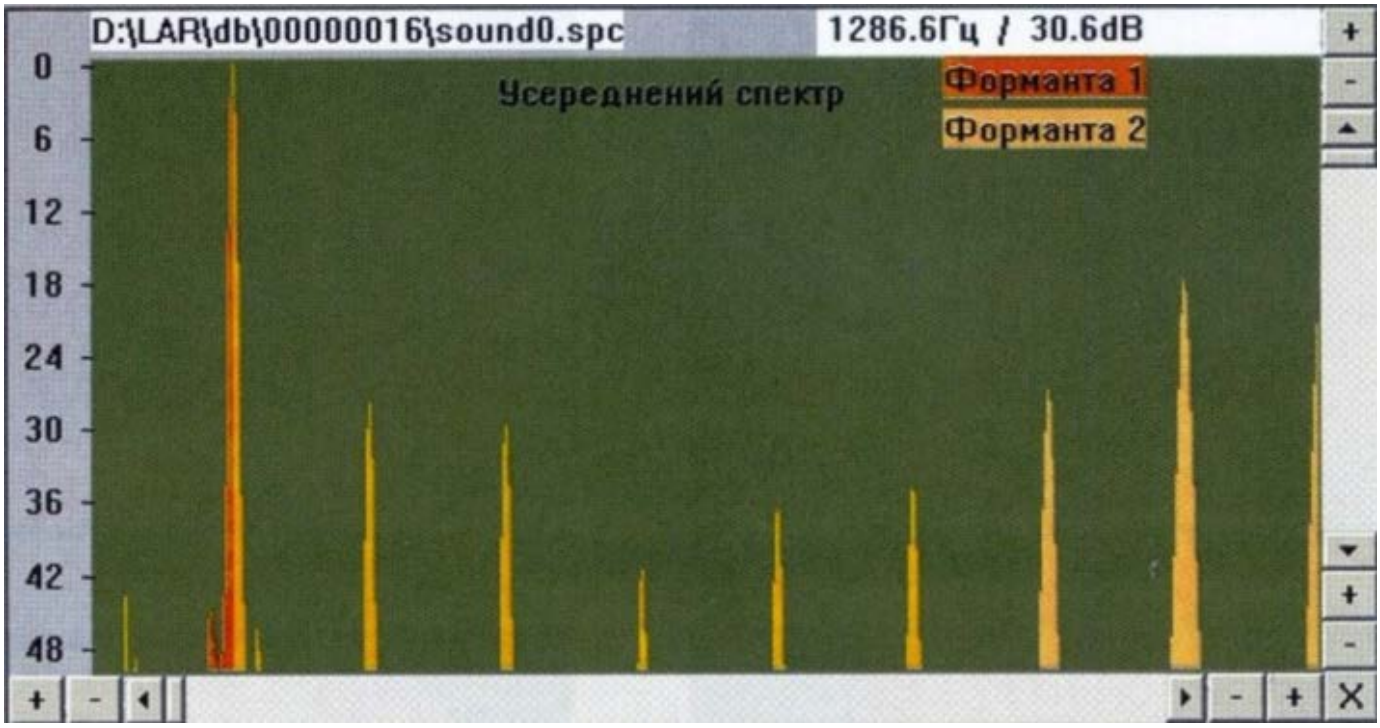

Рис. 1.1.1. Усереднений спектр фонеми «І» у нормі

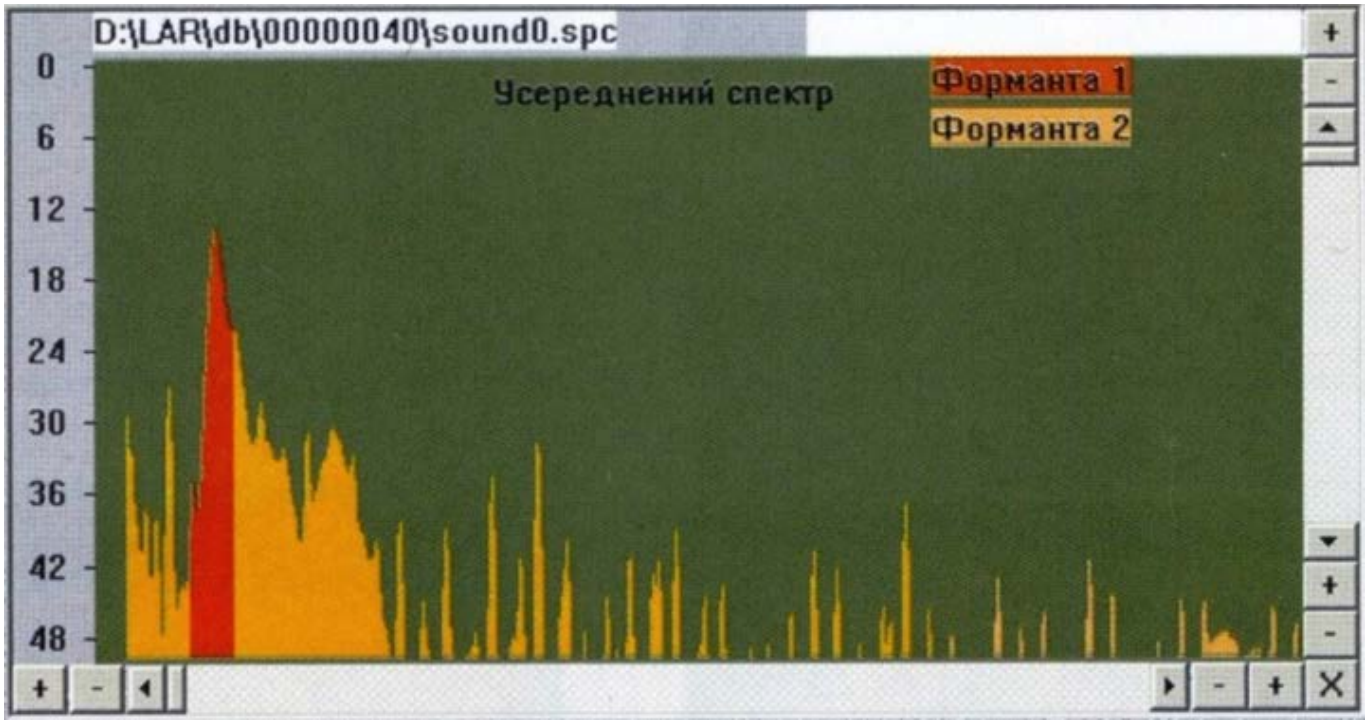

Рис. 1.1.2. Усереднений спектр фонеми «І» при патології голосового апарату (спастична дисфонія)

Автори [4] провели дослідження акустичних характеристик голосу вокалістів-професіоналів зі стажем роботи від 3 до 40 років, які скаржилися на якість голосу при співі, але при стандартному фоніатричному огляді у яких не було виявлено суттєвих відхилень від норми. В якості контрольної групи були обстежені вокалісти-професіонали з нормальним станом голосовою апарату за даними фоніатричного огляду та без скарг на голосову функцію зі стажем від 4 до 43 років. Аналіз отриманих результатів спектрального аналізу голосу дозволив авторам [4] зробити наступні висновки:

1) Вокалісти контрольної групи мають стійкі співочі форманти, багато наповнені обертонами;

2) У обстежених співаків зі скаргами на якість голосу відзначалась менша стійкість формант і бідніше заповнення звуку обертонами;

3) У співаків, які відзначали труднощі при виконанні тих чи інших партій, не завжди фіксувалось співпадіння максимальної кількості формант з серединою їх робочого діапазону, що спостерігалося у осіб контрольної групи.

4) Сонограми низки вокалістів, при задовільній суб'єктивній оцінці голосу і відсутності патологічних змін в гортані, вказували на те, що інтенсивність звуку досягається ними не за рахунок використання резонаторних порожнин, а напругою голосових складок, близькою до граничних. Така напруга небезпечна в плані можливих професійних захворювань голосового апарату;

5) Регулярність піків обертонів у спектрі звуку практично завжди відповідає його якості. Порушення такої регулярності може означати ще не до кінця завершений патологічний процес (наприклад, гіпотонус голосових складок після перенесеного ГРВІ, гострого трахеїту) або початок якогось захворювання, хоча візуально ознаки його могли ще не визначатися;

6) Застосування спектрального аналізу голосу вокалістів дозволяє фоніатру комплексно і повноцінно проводити діагностику і контролювати процес лікування захворювань голосового апарату.

Автори [4] зазначають, що дослідження спектра голосу може виявитися основним діагностичним тестом при діагностиці стійких порушень голосу і, поряд з

дослідженням голосового поля і віброметрією, дозволяє визначити форму голосових порушень у професіоналів. Акустичні тести щодо наявності високої та низької співочих формант у співочому голосі можуть бути високозначущими при визначенні працездатності співака і його професійного прогнозу, можуть слугувати критерієм в діагностиці стійких голосових порушень, а застосування їх на ранніх стадіях професійних захворювань гортані допоможе своєчасно розпочати профілактичні заходи.

1.2.Аналіз наявних програмних рішень

Наразі існує 3 програмні продукти, що можуть допомогти у виконанні задачі спектрального аналізу стану голосового тракту. Перша – це Speech Filing System [3] - це вільне обчислювальне середовище для ПК для проведення досліджень природи мови. Він включає програмні засоби, формати файлів і даних, бібліотеки підпрограм, графіку, спеціальні мови програмування та документацію. Він виконує стандартні операції, такі як отримання, повтор, показ та маркування, спектрографічний та формантний аналіз та фундаментальна частотна оцінка. Він постачається з великим набором готових інструментів для обробки, синтезу та розпізнавання сигналів, а також підтримки вашої власної розробки програмного забезпечення.

Інтерфейс даної програми зображено на рис. 1.2.1:

Як видно з рисунку, даний програмний продукт дає можливість відкрити аудіо файл та провести з ним ряд досліджень. Оскільки відсутня можливість вибору меж аналізу, обраний аудіо-файл має одразу містити лише корисну для користувача інформацію. У випадку лікаря-фоніатра це буде запис максимальної фонації голосної фонеми. На робочому просторі програми відсутній графік спектру сигналу, а якщо його можна якось вивести на головний екран, немає очевидної кнопки, натискаючи яку можна побачити графік спектру. Натомість є певний графік спектрограми, на якій досить чітко виділяються зони формантних областей та їх поведінка у часі.

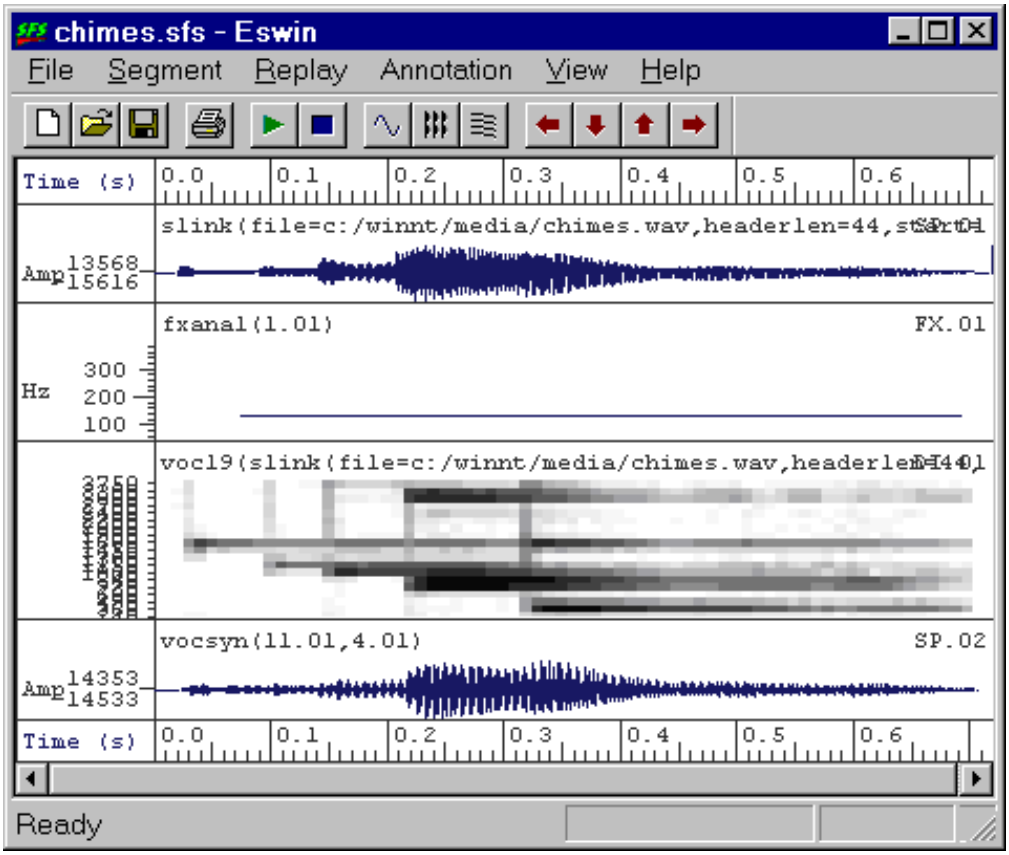

Рис. 1.2.1. Інтерфейс SFS4/Windows

В цілому інтерфейс даного пакету абсолютно не підходить для роботи лікарів. Тут важко зрозуміти, у якій частині вікна що знаходиться та за що відповідає, а сам інтерфейс виглядає застарілим. Зрозуміло, що дане рішення є лише середовищем для розробки подібних систем та потребує значної оптимізації та роботи інженера.

Натомість можна помітити основні можливості, які необхідно виконати при розробці нового продукту. Це:

1. Можливість бачити графік сигналу;
2. Можливість програвати та зупиняти прослуховування аудіо-файлів;
3. Зазначення шляху до файлу для систематизації роботи;
4. Можливість зберегти проведені дослідження у файл, відкривши який можна повернутися до попередніх досліджень.

Другим прикладом є програмний продукт, створений Полом Бурсмою та Девідом Веніком з Університету Амстердама. Це PRAAT [2] - безкоштовний комп'ютерний програмний пакет для аналізу мови в фонетиці. Програма підтримує синтез мовлення, включаючи артикуляційний синтез, а також необхідний нам спектральний аналіз вокальних звуків. На рис. 1.2.2. представлений інтерфейс даного програмного забезпечення.

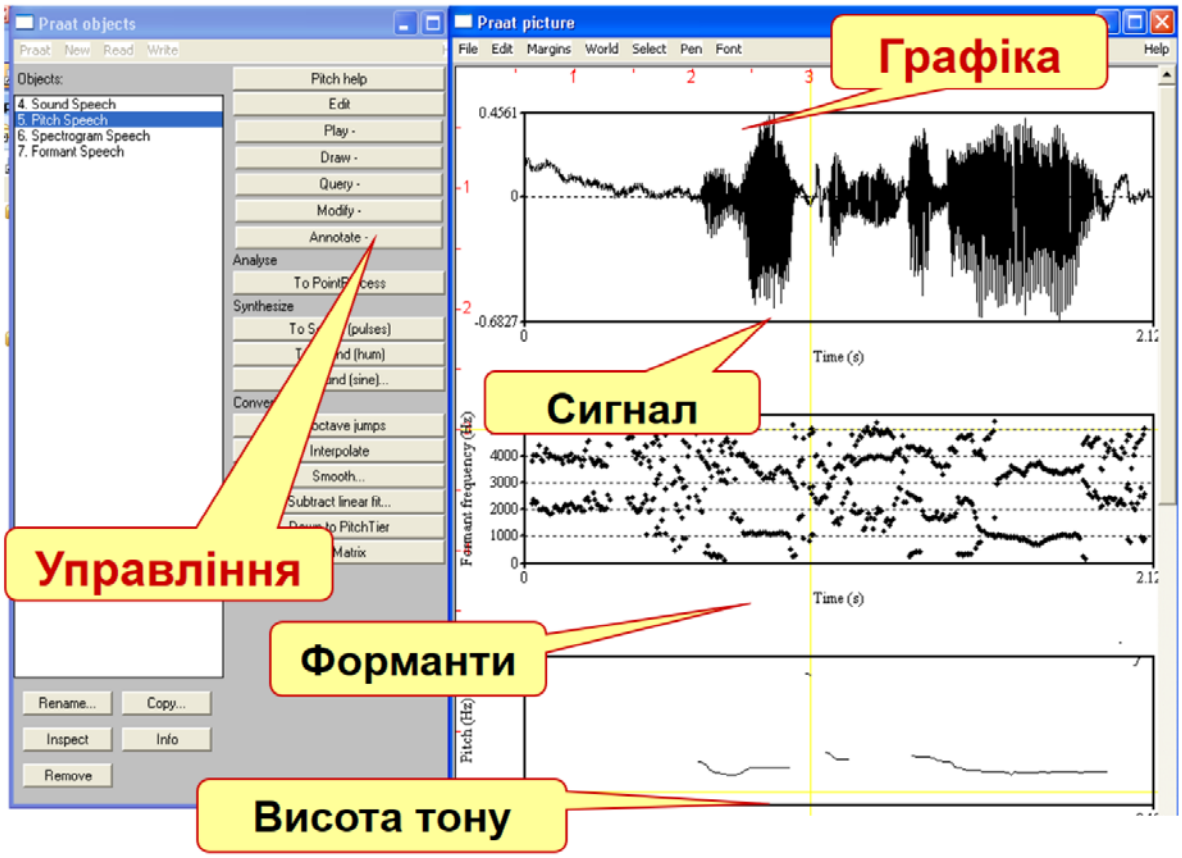

Рис. 1.2.2. Інтерфейс PRAAT

Як можна побачити, існуючий інтерфейс цього застосунку також не придатний для використання лікарем-фоніатром. Звичайно воно має набагато зручніший вигляд, ніж попередній, але все одно цього недостатньо для комфортної роботи користувача.

Перевагою даного інтерфейсу є чітко розділені межі графіків, підписи елементів керування, що значною мірою полегшує роботу користувача. Проте відсутня можливість виставляння візирів та дослідження повноцінного графіку спектру сигналу. Також відсутня можливість обрізання сигналу для вибору із фрази, що складається з набору фонем, одної конкретної фонеми, яку необхідно розглянути. У обох програмах не зображуються літературні формантні області.

Третім програмним рішенням є наведений у джерелі [1] інтерфейс програми комплексного аналізу стану голосового тракту, зображений на рис. 1.2.3.

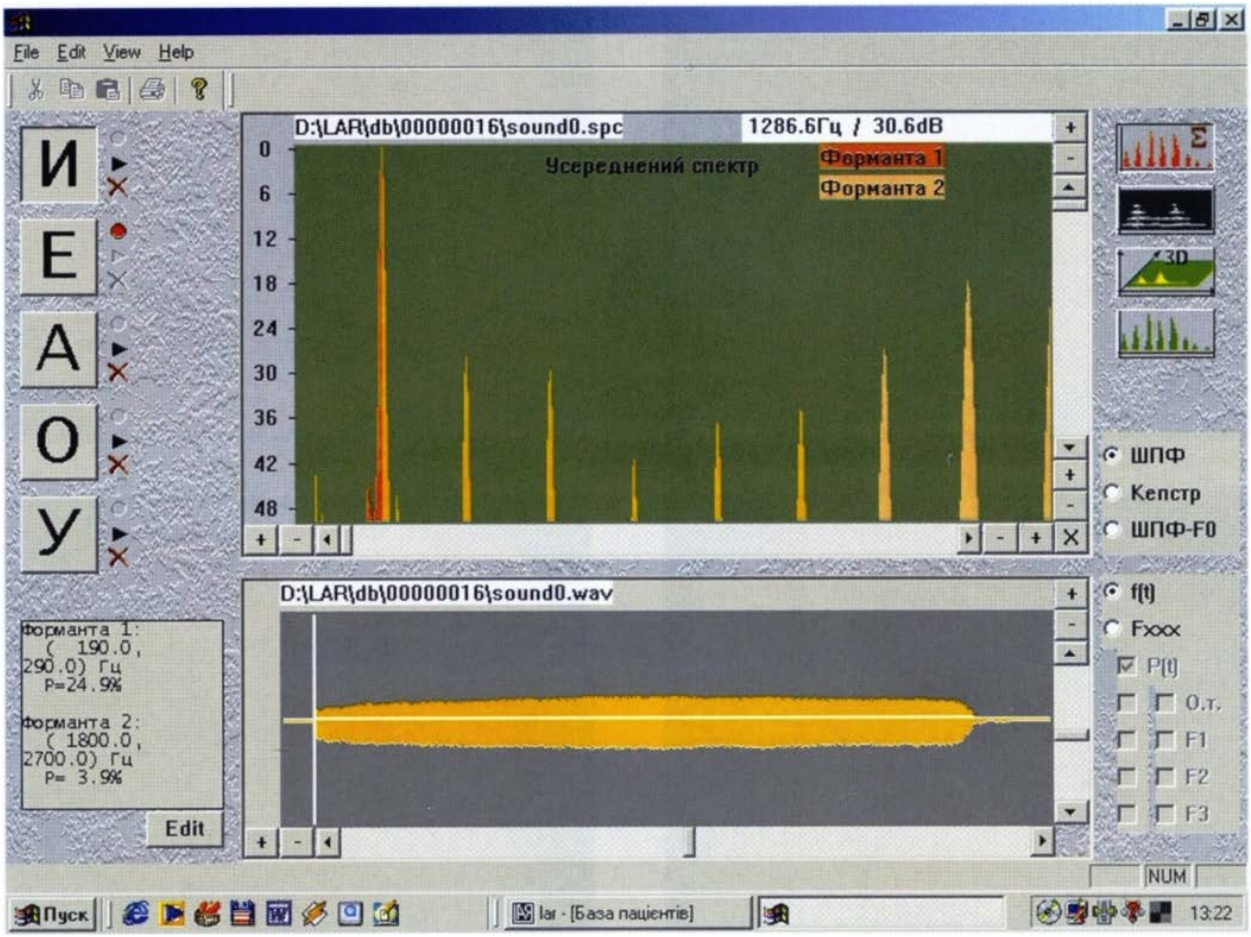

Рис. 1.2.3. Існуючий інтерфейс модуля візуалізації

До переваг даного застосунку можна віднести його адаптованість до роботи лікаря. Тут є можливість почергового запису фонем пацієнтом, а також можливість перемикатися між ними. Кожну з фонем можна окремо прослухати та обрати для аналізу. Також можна побачити візуалізацію формантних областей з літератури на графіку спектру.

Що є дуже важливим, так це можливість бачити абсолютну та відносну потужність спектральних компонент в межах формант.

З огляду на ці два програмні продукти, можна обрати наступні рішення, які необхідно використати при розробці:

1) Можливість бачити графік спектру на головному вікні програми;
2) Зручні та зрозумілі органи керування програмою;
3) Візуалізацію зміни висоти основного тону у часі;

4) Можливість перемикатися між побудовою спектру, спектрограми або кепстру сигналу;
5) Можливість визначення амплітуд спектральних компонент сигналу;
6) Побудову графіку спектрограми для візуалізації мінливості формантних областей з часом;
7) Побудову графіку кепстру для визначення інших необхідних параметрів сигналу, а й відповідно голосу пацієнту.

1.3.Висновки

В даному розділі розглянуто основні можливості, які мають бути в лікаря-фоніатра при роботі з пацієнтом. Також надано наявні програмні рішення та визначено їхні недоліки та переваги.

Наведені дані наголошують на необхідності розробки нової системи та в цілому актуальності даного питання. При тому, що існує низка алгоритмів, що дає змогу виявити патології голосу, лікар-фоніатр потребує зручного та об’єктивного апаратно-програмного інструментарію, що дозволить швидко приступити до лікування.

Опираючись на поставлене технічне завдання, в даній дипломній роботі планується створити власне програмне забезпечення, яке буде відповідати поставленим задачам та вимогам. В основі даної програми буде можливість записувати голосні вокальні звуки та досліджувати їхній спектр, кепстр та спектрограму. Після запису голосних звуків, за отриманими даними мають обчислюватися час максимальної фонації, спектр сигналу, його кепстр та спектрограма.

РОЗДІЛ 2
ВИБІР СЕРЕДОВИЩА ТА МЕТОДІВ РОЗРОБКИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

В даному розділі наведені опис можливостей програмного середовища MATLAB від MathWorks, огляд його пакетів та вибір методу створення інтерфейсу. Крім того висвітлено питання розрахунку спектру сигналу, вибору типів вікон, меж зміни розміру вікон або їх порядку, а також методи аналізу кепстру та спектрограми сигналів.

2.1. Знайомство з MATLAB. Огляд його пакетів інструментів та доповнень.

MATLAB – це середовище та мова технічних інженерних розрахунків, призначене для вирішення широкого спектру питань розробки програмного забезпечення та наукових задач будь-якої складності у всіх сферах. MATLAB об’єднує в собі:

1) Мову інженерних розрахунків;
2) Графічні додатки (робота з графічним інтерфейсом);
3) Засоби розробки програмного забезпечення;
4) Більше сотні прикладних програм (так званих toolboxes), що виступають як професійне розширення та адаптація системи під рішення певних класів математичних і наукових-технічних завдань.

MATLAB є зручним засобом для вирішення математичних задач, базуючись на великій кількості функцій аналізу даних. До них відносять:

1) Матриці та лінійна алгебра – робота з даними у векторному або матричному представленні, вирішення лінійних рівнянь;
2) Многочлени – знаходження коренів многочленів, операції над ними та їх диференціювання, інтерполяція та екстраполяція кривих;
3) Математична статистика та аналіз даних – статистичні функції, цифрова фільтрація, швидке перетворення Фур’є;

4) Обробка даних – набір різноманітних функцій, а також побудова графіків, пошук нулів рівняння, чисельне інтегрування;

Значною перевагою програмного середовища MATLAB є його робота з графіками. Оскільки задача даної дипломної роботи базується на побудові графіків спектру, розробка програмного продукту повністю базується на можливості обраного середовища візуалізувати дані.

У пакет MATLAB входить велика кількість доповнень, що допомагає в аналізі аудіо файлів. До таких задач можна віднести:

1) Введення у середовище масиву даних з аудіо файлів формату .WAV або .MP3, з визначенням частоти дискретизації та розрядності;
2) Візуалізацію сигналу;
3) Створення, аналіз та використання різноманітних цифрових рекурсивних та нерекурсивних фільтрів;
4) Визначення амплітуди, фази, групової затримки та імпульсної характеристики отриманих фільтрів;
5) Оцінка спектру сигналу, його спектрограми та кепстру.

Одним з таких доповнень є Signal Processing Toolbox. Він надає функції та програми для аналізу, попередньої обробки та вилучення частин з рівномірно та нерівномірно дискретизованих сигналів. Пакет інструментів включає інструменти для проектування та аналізу фільтрів, зміни частоти дискретизації, згладжування, видалення постійної складової та оцінки спектру потужності сигналу. Крім того є функції визначення ознак сигналів, його огинаючих, знаходження піків та ознак сигналів, оцінка схожості сигналів, а також параметрів відношення сигнал/шум, або коефіцієнту спотворень.

За допомогою програми Signal Analyzer можна попередньо обробляти та аналізувати декілька сигналів одночасно у часових, частотних та частотно-часових областях без написання коду; досліджувати довгі сигнали та витягати з них корисні області. За допомогою програми Filter Designer можна створювати та аналізувати цифрові фільтри та аналогові фільтри, включаючи фільтри Баттерворта, Чебишева, Бесселя та еліптичний.

Signal Processing Toolbox також дозволяє проводити аналіз сигналів частотно-часовими техніками, такими як спектрограма, та оцінювати зміну вигляду спектру у часовій області та оцінювати степінь схожості сигналів с спектральній області, вимірюючи когерентність спектрів.

Інтерфейс даної програми наведений на рис. 2.1.1:

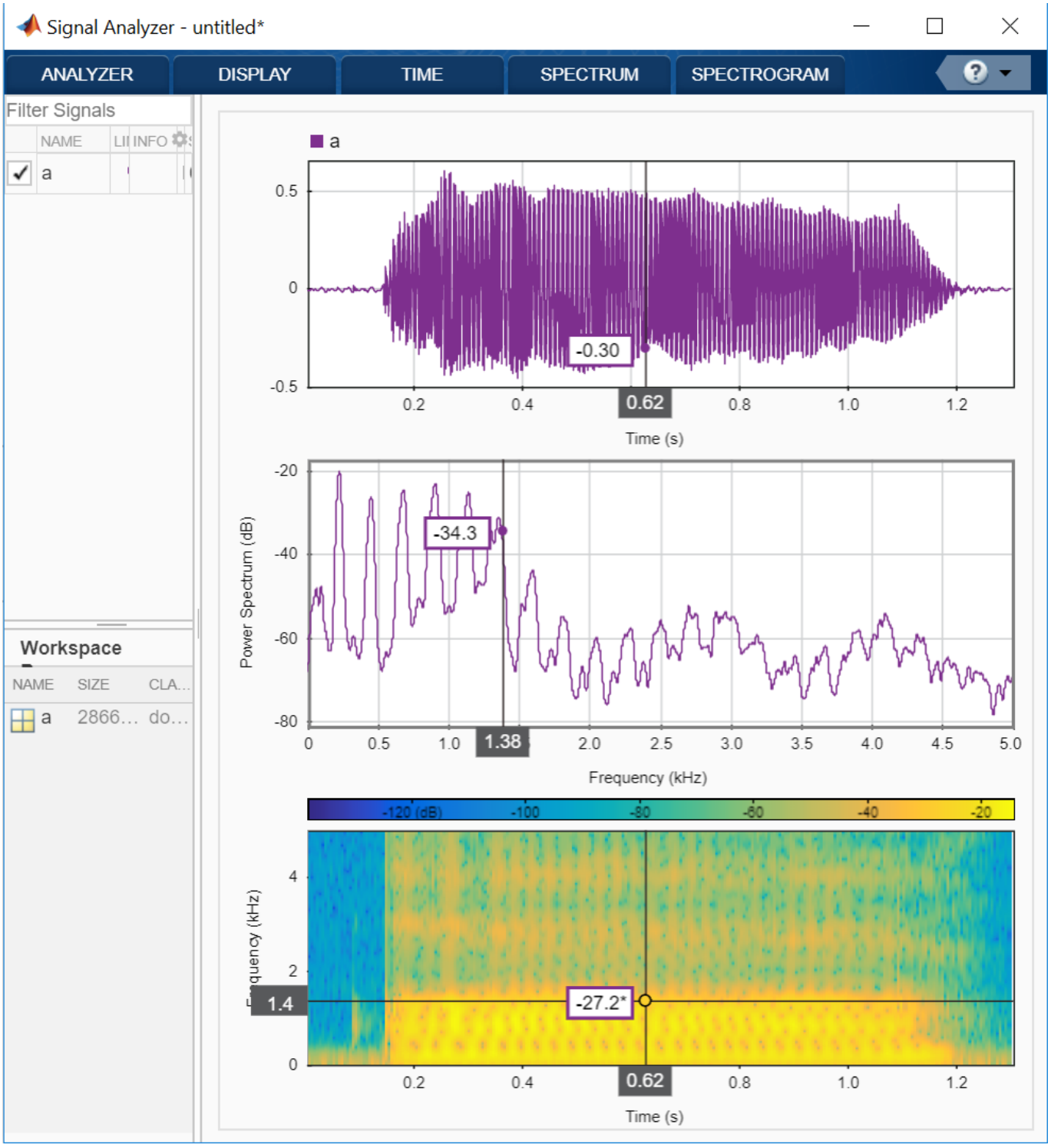

Рис. 2.1.1. Інтерфейс Signal Analyzer

Цей програмний продукт демонструє можливий вигляд майбутнього розробленого інтерфейсу. Проте в цілому сам по собі він підходить лише для дослідницьких та інженерних цілей, оскільки даний застосунок не дає доступу до збережених аудіо файлів, а працює лише з внесеними у блок Workspace даними. Через це необхідно імпортувати дані через командне вікно MATLAB.

Для реалізації цієї задачі існує пакет Audio Toolbox, що надає інструменти для обробки звуку, аналізу мови і акустичних вимірювань. Він включає в себе алгоритми обробки аудіосигналу (такі як вирівнювання і управління динамічним діапазоном) і акустичного вимірювання (такі як оцінка імпульсної характеристики, октавна фільтрація і перцептивне зважування). Він також надає алгоритми для вилучення аудіо і мовних функцій (наприклад, MFCC і pitch) і перетворення аудіосигналу.

Додатки в складі продукту підтримують тестування алгоритмів в реальному часі, вимірювання імпульсної характеристики і маркування аудіосигналу. Продукт надає потокові інтерфейси для звукових карт ASIO, WASAPI, ALSA і CoreAudio і MIDI-пристроїв, а також інструменти для створення і розміщення стандартних аудіо плагінів, таких як VST і Audio Units.

За допомогою Audio Toolbox можна імпортувати, маркувати і змінювати набори аудіо, а також оцінювати параметри (діагностичні ознаки) і перетворювати сигнали для машинного навчання і глибокого навчання. Також можна прототипувати алгоритми обробки звуку в режимі реального часу шляхом потокової передачі звуку з низькою затримкою при налаштуванні параметрів і візуалізації сигналів. Можна перевірити свій алгоритм, перетворивши його в аудіо-плагін для запуску в зовнішніх додатках, таких як Digital Audio Workstations (DAWs). Плагін хостинг дозволяє використовувати зовнішні аудіо плагіни, як звичайні об'єкти для обробки масивів в MATLAB. Підключення звукової карти дозволяє виконувати виміри на реальних аудіосигналах і акустичних системах.

Audio Toolbox реалізує можливість підключення до стандартних аудіо драйверів. Це дає можливість підключення до стандартних звукових карт ноутбуків або настільних ПК для потокової передачі багатоканального звуку з малою затримкою між будь-якою комбінацією форматів файлів та живими входами і

виходами аудіоінтерфейсу. Таким чином ми можемо зчитувати аудіо дані з входів звукової карти, виконувати їх аналіз, обробку, зберігання та відтворення.

Наступним питанням є створення користувацького інтерфейсу програми. MATLAB пропонує 3 різні способи створення програм:

1) Використовуючи функції MATLAB для створення інтерфейсів програмно;
2) Використовуючи пакет GUIDE;
3) Використовуючи пакет App Designer;

Кожен з цих підходів пропонує різний робочий процес і дещо різний набір функціональних можливостей. Найкращий вибір залежить від вимог проекту та від того, якому способу розробник віддає перевагу.

Перший підхід базується на ручному написанні скрипту, кодуючи компонування та поведінку додатка, використовуючи бібліотеку функцій MATLAB. При такому підході створюється традиційне графічне вікно, де розміщуються інтерактивні компоненти. Ці програми підтримують ті ж типи графіки та інтерактивних компонентів, які підтримує GUIDE. Такий підхід є досить клопітним та незручним.

Інший підхід базується на використанні пакету GUIDE – середовище, створення інтерфейсу в якому базується на drag-and-drop підході, коли користувач перетягує інтерактивний компонент з бібліотеки компонентів на робоче поле майбутнього застосунку. Такий метод є зручним для користувача, оскільки не потребує написання та редагування коду при розміщенні необхідних компонентів на сітці інтерфейсу. Проте цей метод потребує окремого оформлення та написання callback-функцій, які відповідають за відгук компонентів на зміну їх параметрів (наприклад натискання push-button, або вибір елементу з listbox). Інтерфейс GUIDE наведений на рис. 2.1.2. Добре видно, що інтерфейс виглядає дещо застарілим. Крім того він не має чітко виділеної зони редагування параметрів компонентів. Прописування коду програми відбувається у окремому вікні, що змушує постійно перемикатися вікнами. Бібліотека компонентів мала, компоненти мають порівняно непривабливий вигляд. З 2019-го року пакет GUIDE не підтримується MATLAB і буде видалений з майбутніх оновлень, хоча існує можливість міграції застосунків до пакету App Designer.

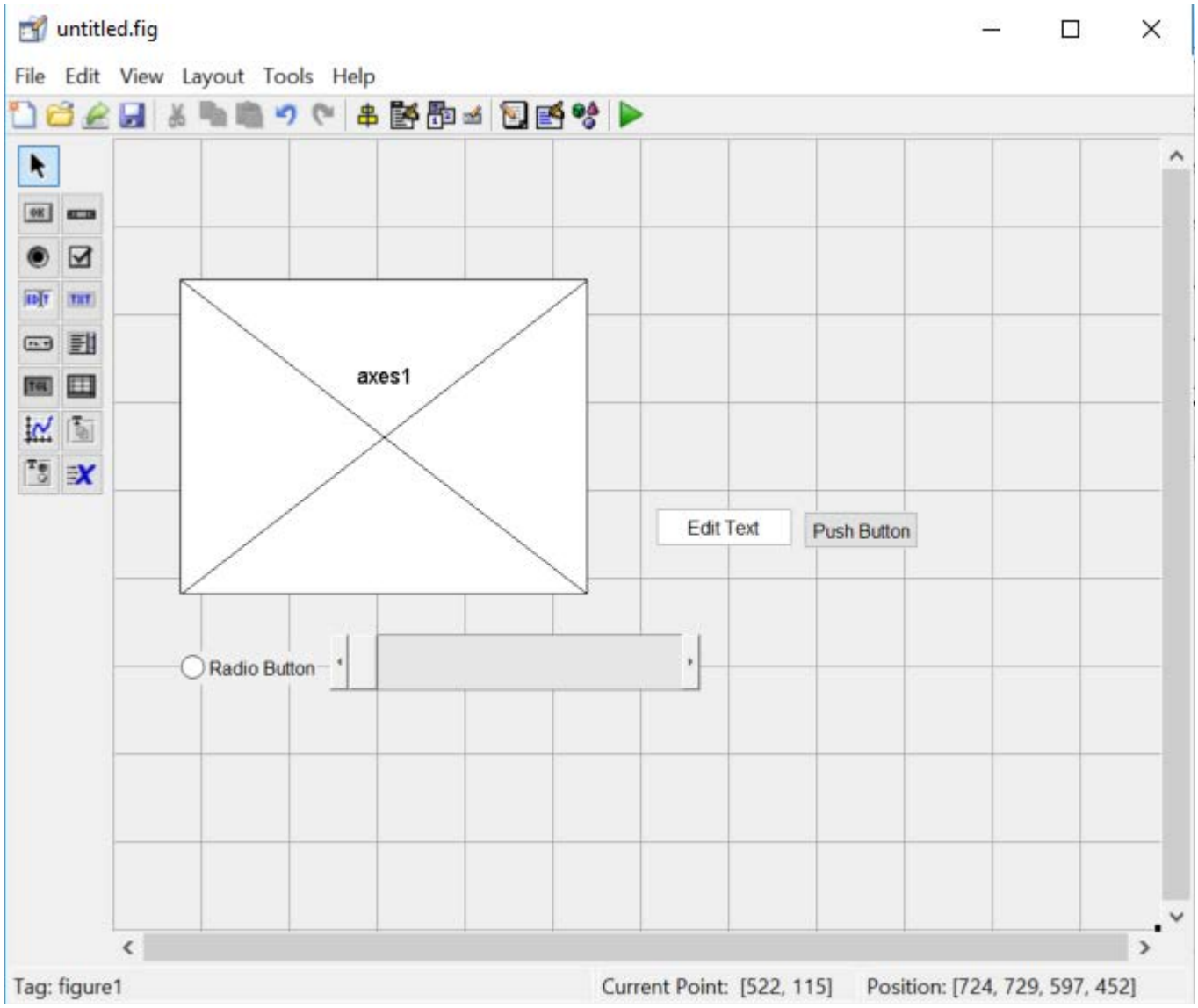

Рис. 2.1.2. Інтерфейс GUIDE

Найкращим та найбільш адаптованим для розробника є пакет App Designer. Він дозволяє створювати професійні програми навіть тим користувачам, які не є професійними розробниками програмного забезпечення. Щоб створити дизайн графічного інтерфейсу користувача (GUI), використовується схожа до GUIDE методика drug-and-drop, за допомогою якої макетування відбувається дуже швидко та зручно. App Designer інтегрує два основних завдання побудови додатків - викладення візуальних компонентів графічного інтерфейсу користувача та спостереження поведінки коду програми з можливістю редагування коду в вікні самого застосунку рис.2.1.3. Це рекомендоване середовище для створення програм у MATLAB.

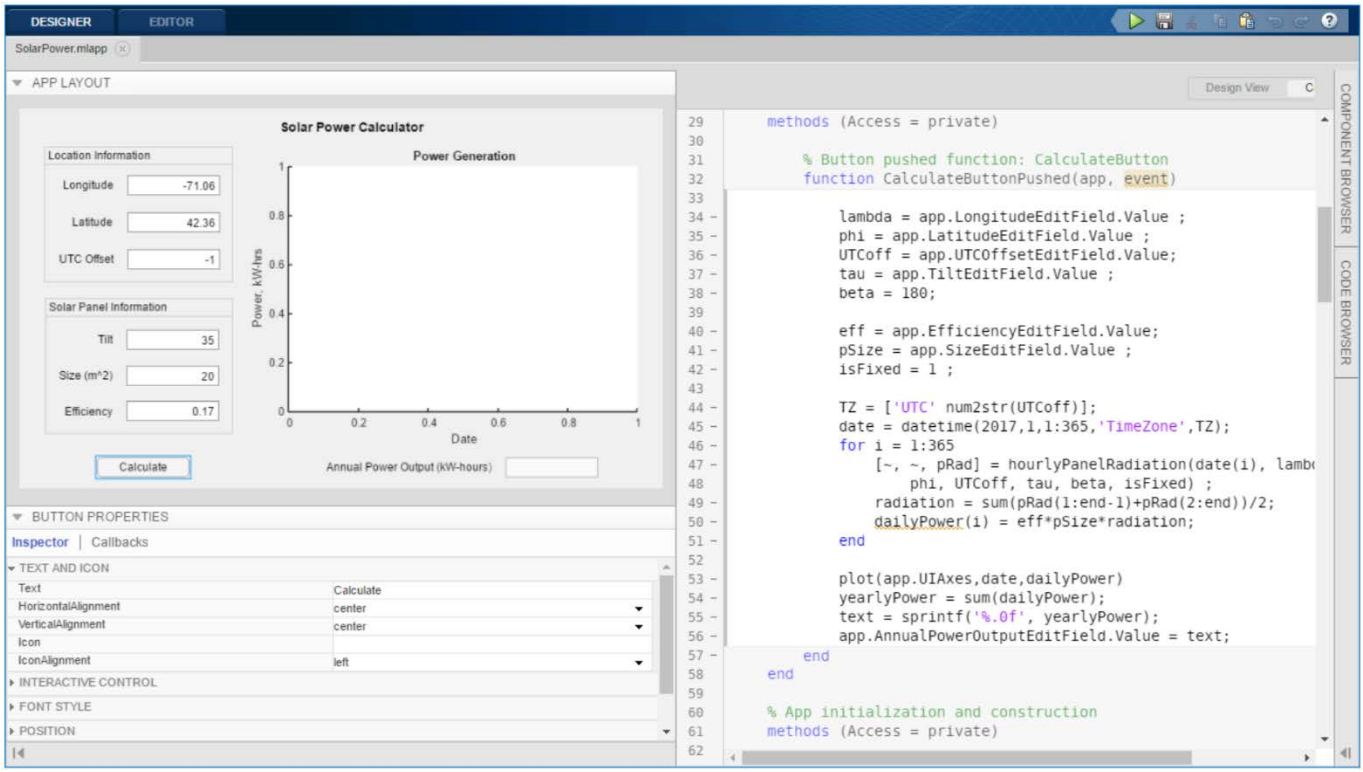

Рис. 2.1.3. Інтерфейс програми та поля редагування коду в одному вікні

Значною перевагою доданку App Designer є наявність сітки та вирівнювання компонентів по ній. Крім того існує можливість прив’язування та групування елементів один з одним. Згруповані компоненти можуть разом переміщуватись, не змінюючи власного положення один відносно одного, що є дуже важливим для правильного макетування інтерфейсу застосунку (рис. 2.1.4).

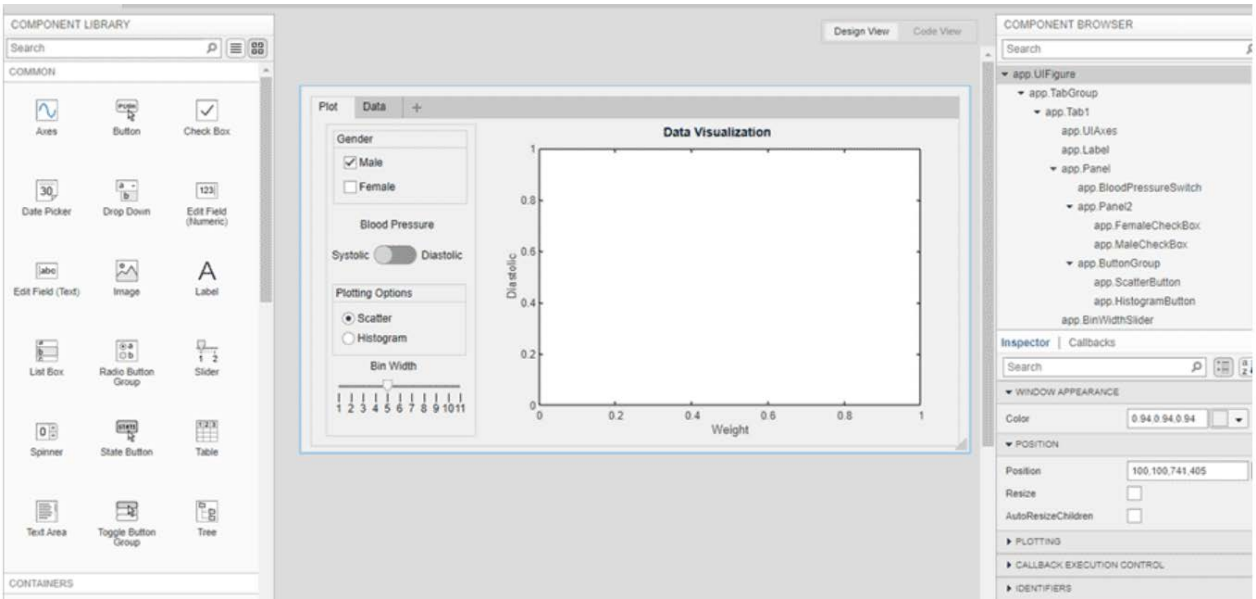

Рис. 2.1.4. Демонстрація можливостей групування компонентів на робочому просторі

Ще одною перевагою є наявність браузеру компонентів та поля редагування їх параметрів. Це полегшує створення функцій callback та редагування вигляду кожного компоненту. До редагованих параметрів відносяться вставлення зображень до компоненту та його прив'язка, редагування тексту, розміру, шрифту та прив'язки тексту на полі компоненту. Вигляд даних рішень показано на рис. 2.1.5:

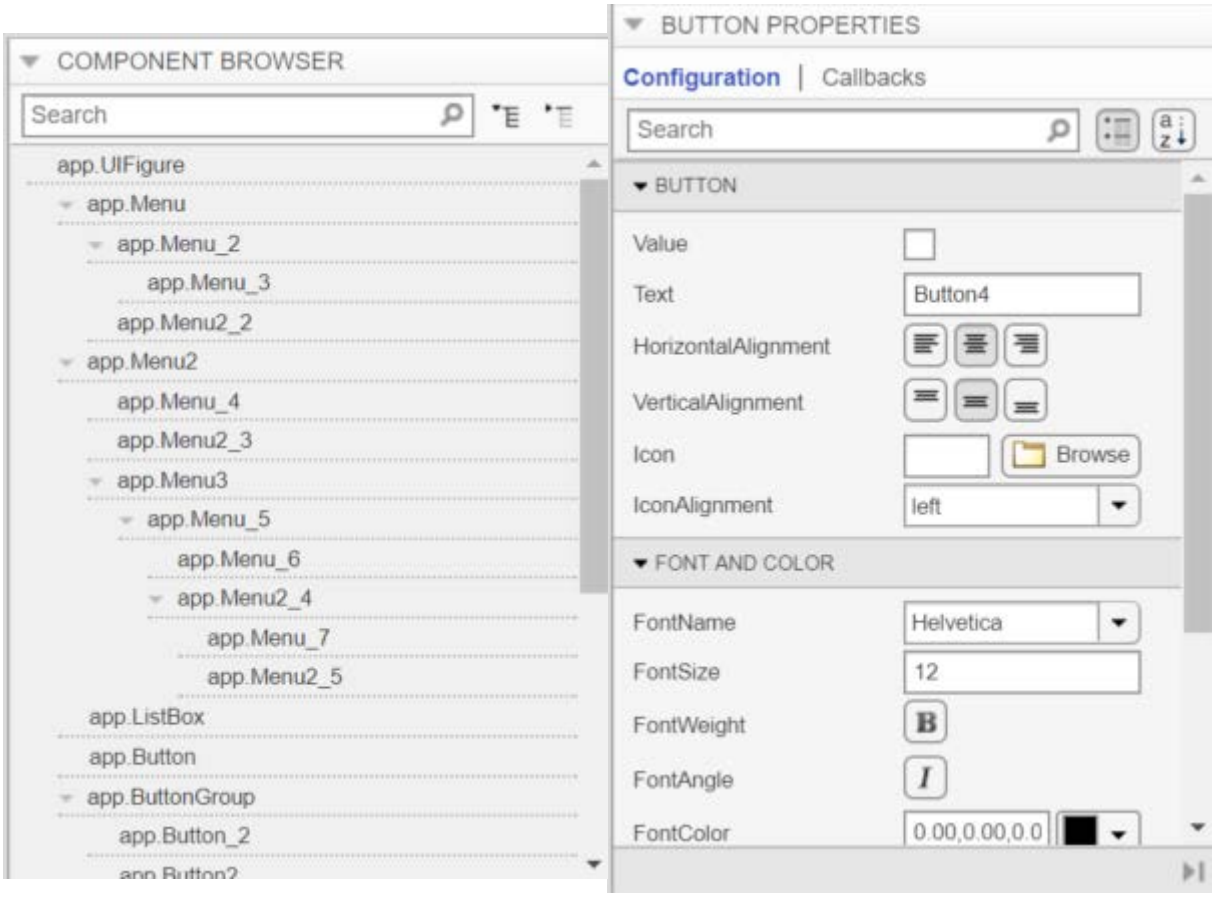

Рис. 2.1.5. Вигляд браузеру компонентів та поля редагування параметрів

Програми створюються зі стандартними компонентами, такими як кнопки, прапорці, дерева та спадні списки. App Designer також пропонує такі елементи керування, як датчики, світильники, ручки та вимикачі, які дозволяють повторити зовнішній вигляд та дії панелей приладів. Користувач також може використовувати компоненти контейнерів, такі як вкладки, панелі та макети сітки для організації свого інтерфейсу користувача. Досить широка бібліотека компонентів зоброжена на рис. 2.1.6.

До кожного компоненту можна додати callback-фунцію, яка реагує на кожну взаємодію миші та клавіатури користувачем з компонентом. Таким чином можна

змінювати різноманітні параметри графіків або, наприклад, значень функцій, що будуються на полі Axes.

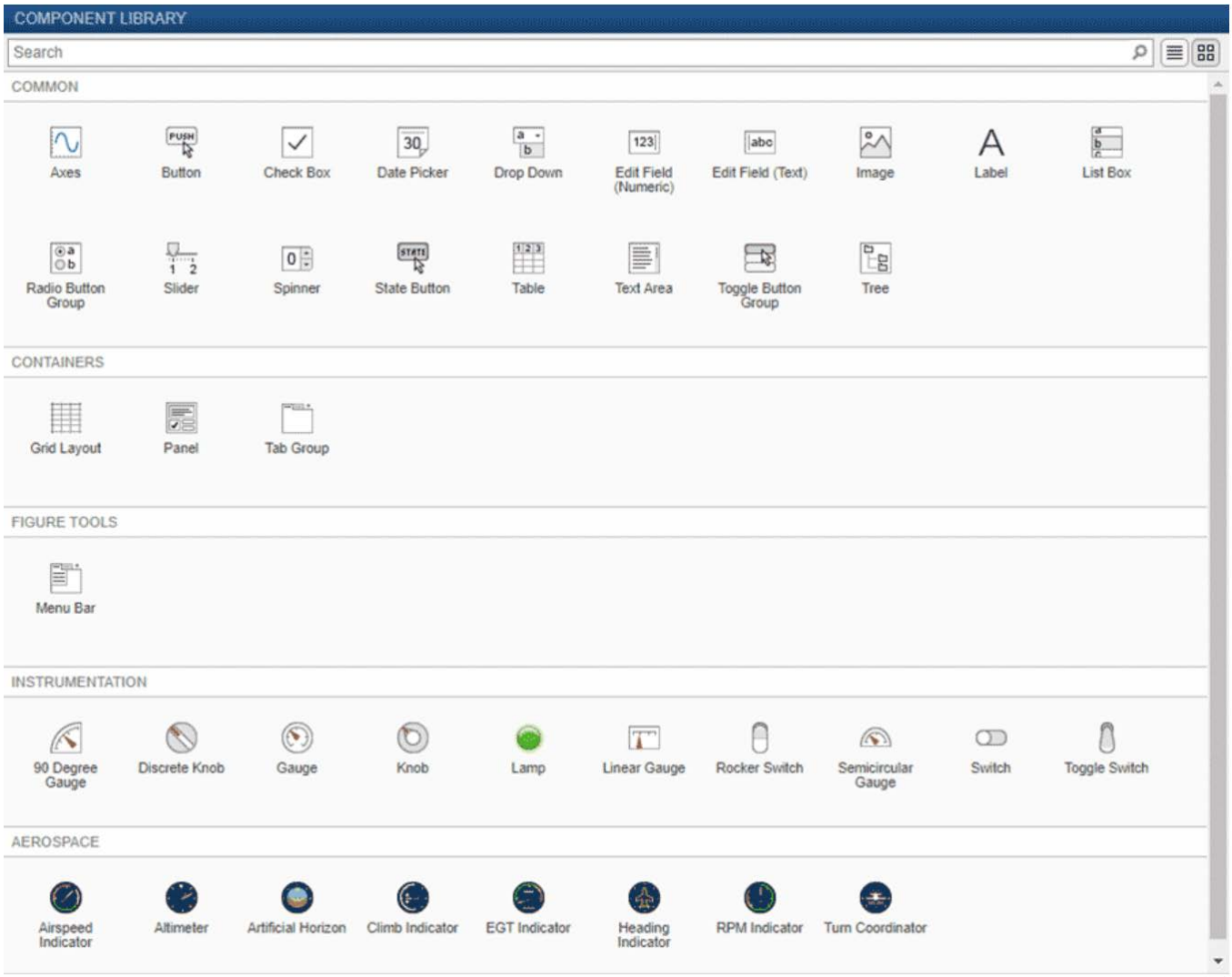

Рис. 2.1.6. Бібліотека компонентів App Designer

Останнім питанням, що пов’язане з розробкою програмного забезпечення на базі середовища MATLAB, є компіляція отриманого коду в EXE-файл. Бібліотеками MATLAB надаються два інструменти: MATLAB Coder та MATLAB Compiler.

MATLAB Coder генерує код C/C++ з алгоритмів коду MATLAB для різних апаратних платформ, від настільних систем до вбудованих платформ. Підтримується велика частина мови MATLAB і широкий набір додаткових бібліотек. Підтримується генерація як вихідного коду, так і статистичних і динамічних бібліотек. Згенерований код можливо зчитати і переносити. Успадкований код і бібліотеки можуть бути підключені до MATLAB, щоб отримати максимальну ефективність для ключових частин алгоритму або щоб повторно використовувати раніше створені

алгоритми. Для прискорення розрахунків алгоритмів, розроблених в MATLAB можна згенерувати MEX-функцію, що виконується в MATLAB.

Embedded Coder розширює MATLAB Coder і підтримує налаштування коду, специфічні для конкретної цільової платформи, створення платформо-залежного коду, а також верифікації коду в режимах SIL і PIL.

MATLAB Compiler дозволяє розробляти в MATLAB як автономні програми, так і веб-програми. За допомогою компілятора MATLAB також можна пакувати та розгортати програми MATLAB як додатки великих даних MapReduce та Spark та як додатки Microsoft Excel. Кінцеві користувачі можуть запускати програми, використовуючи MATLAB Runtime.

Більш того, можна здійснювати хостинг розроблених в MATLAB веб додатків за допомогою MATLAB Web App Server, що поставляється разом з MATLAB Compiler. За допомогою MATLAB Compiler SDK можна створювати з програм на мові MATLAB модулі для інтеграції з іншими мовами програмування.

MATLAB Compiler дає можливість створювати незалежні додатки з MATLAB алгоритмів і програм для вільного поширення без необхідності придбання ліцензії або встановлення середовища MATLAB на комп'ютер користувача. Сформовані з алгоритмів MATLAB незалежні додатки є повноцінними програмами, які можуть бути укомплектовані графічним або консольним інтерфейсом, розробленим за допомогою App Designer.

MATLAB Compiler дає доступ до використання інтерактивних та інтуїтивних засобів для компоновки MATLAB розробок в незалежні застосунки. Також можна створювати повноцінні інсталятори для розроблених застосунків з відомостями про автора, заставками та даними про версію програми. У процесі створення інсталятора Compiler допомагає визначити усі необхідні файли та залежності, щоб провести зборку, приклад якої наведено на рис 2.1.7.

MATLAB Compiler та Coder працюють одночасно, але дають різний результат. Великою перевагою використання пакету Coder є те, що на виході ми отримуємо C/C++ вихідний код, що дає можливість роботи на великій кількості платформ, і що головне, в залежності від алгоритмів можлива більша швидкодія готового продукту.

Також згенерований код не потребує жодних додаткових бібліотек для його функціювання.

До його недоліків варто віднести те, що не всі бібліотеки та відповідно функції підтримуються та можуть бути реалізовані у вихідному коді. Також Coder гарантовано не підтримує MATLAB графіки та більшості аудіо, відео та фото форматів.

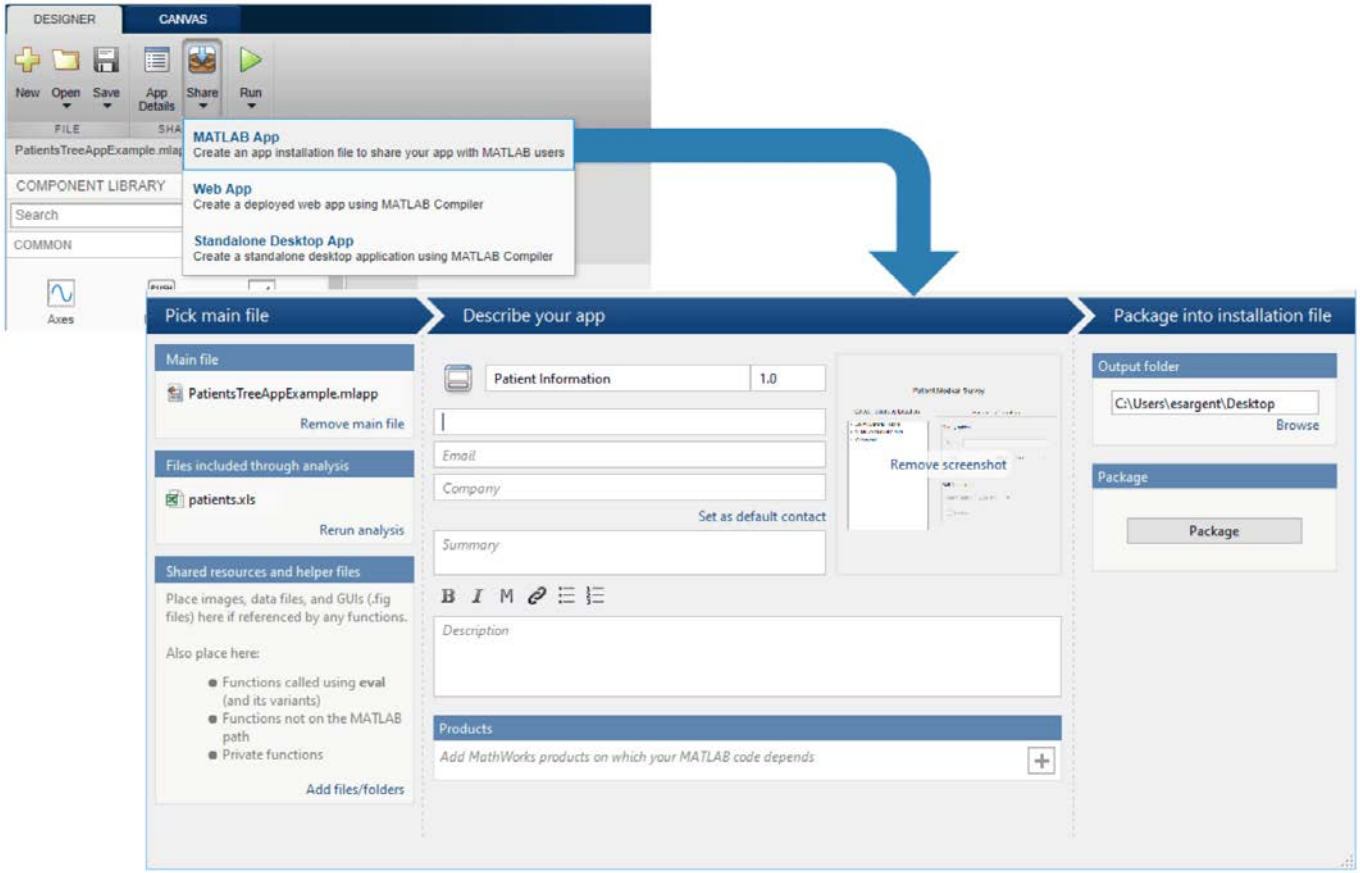

Рис. 2.1.7. Створення інсталятору для поширення користувачам

Перевагою ж MATLAB Compiler є отримання на виході EXE-файлу. Крім того він повністю підтримує більшість доданків MATLAB. Слідом за цим йде повна підтримка MATLAB графіки та всіх форматів даних, що підтримуються самим MATLAB.

Важливими недоліками даного підходу є його потенційно мала швидкодія, а також закритість вихідного виконуючого коду. Для введення змін необхідно мати початковий проект, що містить код MATLAB, а потім знову його пакувати.

Головним недоліком MATLAB Compiler є необхідність встановлювати разом з застосунком досить велику бібліотеку.

MATLAB Runtime - це окремий набір спільних бібліотек, що дозволяє виконувати складені програми або компоненти MATLAB. Якщо вони використовуються разом, MATLAB, MATLAB Compiler та MATLAB Runtime дозволяють швидко та надійно створювати та розповсюджувати програми або компоненти програмного забезпечення. Дана бібліотека може поширюватися як разом з запакованим інсталятором програми, так і встановлюватись окремо з офіційного сайту MATLAB. Важливо, щоб версія Runtime, на якій було розроблено програмне забезпечення та версія, встановлена на пристрої користувача співпадали.

2.2. Методи оцінки спектру сигналу

Програмне розширення Signal Processing Toolbox від MATLAB пропонує різні методи аналізу спектру сигналу. Вони базуються на ДПФ – дискретному перетворенню Фур'є. Для пришвидшення розрахунків використовуються алгоритми ШПФ – швидкого перетворення Фур'є.

Всі методи дослідження поділяються на два класи:

1) Непараметричні (періодограма та метод Уелча);

2) Параметричні (авторегресійна модель, MUSIC та EV).

Більшість досліджуваних сигналів швидкими методами Фур'є є випадковими процесами. В силу цього при звичайному аналізі будь-якого аудіо сигналу ми отримуємо спектр лише єдиної реалізації процесу. Методи, що використовують лише дані з самого вхідного сигналу звуться непараметричними.

Непараметричним є метод періодограми – оцінка спектральної щільності потужності, що отримується по N відлікам єдиної реалізації випадкового процесу, тобто звичайного запису аудіо. Розрахунок періодограми відбувається за наступною формулою:

$$\hat{W}(\omega) = \frac{1}{Nf_{\partial}} \left| \sum_{k=0}^{N-1} x(k) e^{-j\omega kT} \right|^2 \qquad (2.1)$$

Тут $x(k)$ - це послідовність відліків сигналу. Для оцінки спектральної щільності потужності аналогового сигналу, що відновлюється за відліками $x(k)$, необхідне ділення на частоту дискретизації $f_д$.

При розрахунку спектральної щільності потужності може використовуватися вагова функція. В якості вагової функції використовується вікно з коефіцієнтами $w(k)$, тоді оцінка називається модифікованою періодограмою та розраховується за формулою:

$$\hat{W}(\omega) = \frac{1}{f_д} \frac{\left| \sum_{k=0}^{N-1} x(k) w(k) e^{-j\omega kT} \right|^2}{\sum_{k=0}^{N-1} |w(k)|^2} \tag{2.2}$$

Дисперсія оцінки спектральної щільності потужності методом періодограми досягає квадрату її математичного очікування незалежно від кількості відліків N частини сигналу, що оброблюється.

З ростом кількості досліджуваних відліків, значення періодограми все більше флуктують так, що графік такої оцінки стає все більш різаним. Вплив цього ефекту можна бачити на рис. 2.2.1:

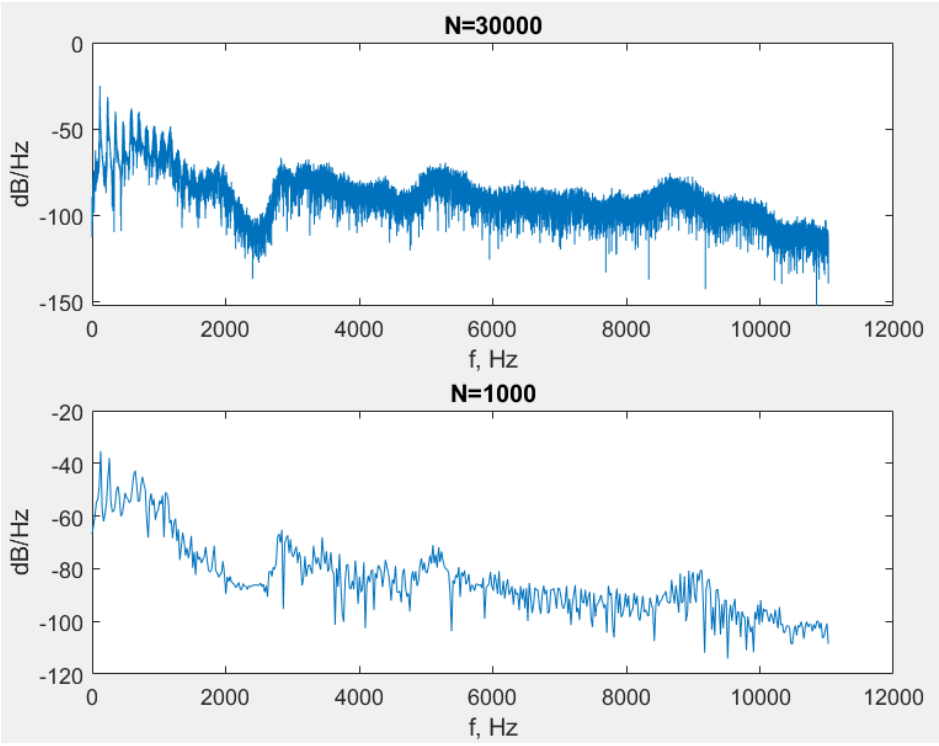

Рис. 2.2.1. Вигляд оцінки спектру методом періодограми при різних довжинах сегменту

Побудова:

```
[a,sf]=audioread('C:\Users\Пользователь\Sounds\Example.wav');
a1=a(73000:73000+30000);
a2=a(73000:73000+1000);
[Pxxx1, f1] = periodogram(a1,[],[],sf);
[Pxxx2, f2] = periodogram(a2,[],[],sf);
figure(1);
subplot(2,1,1), plot(f1, 10*log10(Pxxx1)); title('N=30000'), xlabel('f, Hz'),
ylabel('dB/Hz')
subplot(2,1,2), plot(f2, 10*log10(Pxxx2)); title('N=1000'),  xlabel('f, Hz'),
ylabel('dB/Hz')
```

Для зменшення цього ефекту необхідно вдатися до певного усереднення результату. Зазвичай пропонується згладжувати швидкі флуктуації отриманого спектру шляхом усереднення за сусідніми частотами. Даний метод реалізується шляхом обчислення згортки періодограми з функцією, що виконує згладжування.

Іншим методом усереднення є метод Бартлетта. За цим методом аналізований сигнал поділяється на фрагменти, що не мають перекриття. Тоді для кожного фрагменту вираховується періодограма, і тоді отримані значення усереднюються.

За Уелчем до останнього методу вводяться поліпшення: використання вагової функції та сегментів з перекриттям. Вагова функція призводить до зменшення розтікання спектру. Завдяки введенню перекриття ми отримуємо збільшення кількості сегментів та зменшення в силу цього дисперсії оцінки, яка зменшується приблизно пропорційно кількості сегментів. Порівняння стандартного методу періодограми та методу Уелча зображено на рис. 2.2.2.

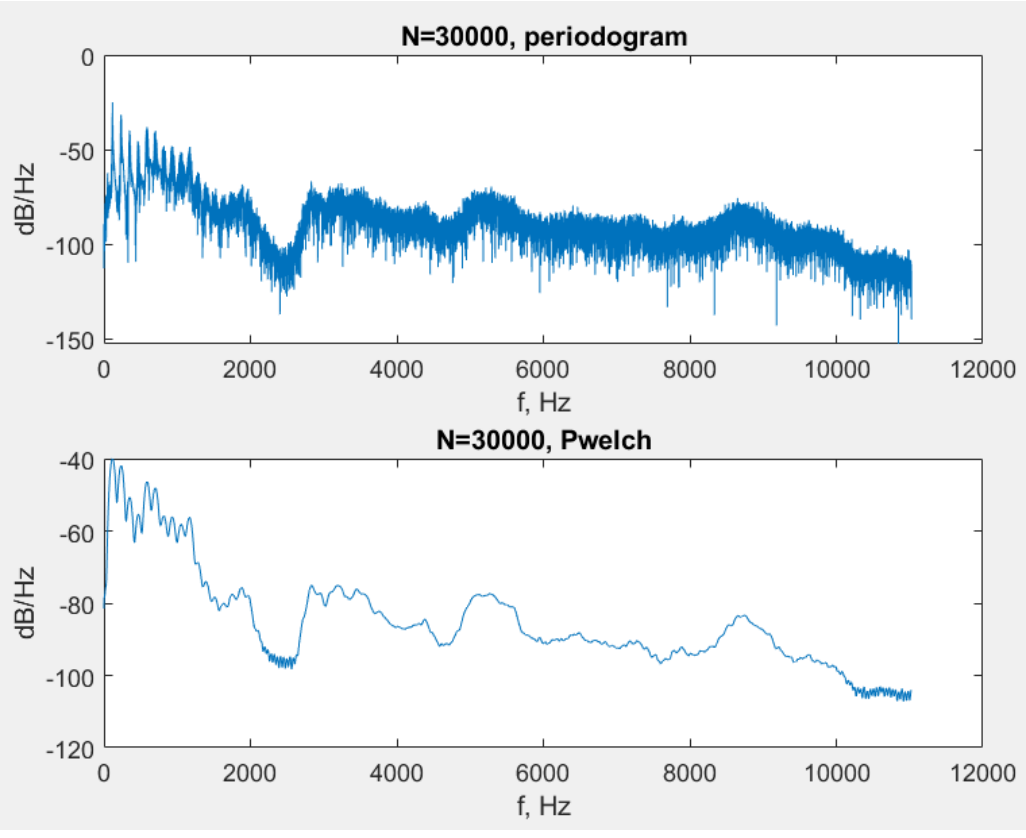

Рис. 2.2.2. Оцінка спектру методом періодограми та Уелча

Побудова:

```
a1=a(73000:73000+30000);
nfft=512;
win1 = hamming(nfft);
[Pxxx1, f1] = periodogram(a1,[],[],sf);
[Pxxx2, f2] = pwelch(a1,win1,nfft/2,sf/2,sf);
figure(1);
subplot(2,1,1), plot(f1, 10*log10(Pxxx1)); title('N=30000, periodogram'),
xlabel('f, Hz'), ylabel('dB/Hz')
subplot(2,1,2), plot(f2, 10*log10(Pxxx2)); title('N=30000, Pwelch'),  xlabel('f,
Hz'), ylabel('dB/Hz')
```

Алгоритм оцінки спектру за методом Уелча виглядає наступним чином:

1) Перелік відліків дискретного сигналу поділяється на сегменти з перекриттям;

2) Усі сегменти множаться на обрану вагову функцію;

3) Для отриманих сегментів обраховуються періодограми;

4) Отримані модифіковані періодограми усереднюються.

Пакет Signal Processing від MATLAB містить цілий ряд стандартних вагових функцій. Вони повертають вектори відліків, що можуть бути використаними в якості одного з параметрів різних функцій непараметричного аналізу спектру сигналу.

Іншою групою методів є параметричні. Вони базуються на використання певної математичної моделі, що відповідає досліджуваному випадковому процесу. При цьому сам спектральний аналіз зводиться до вирішення задачі оптимізації параметрів моделі, знаходячи такі, при яких сама модель буде максимально наближеною до реального сигналу.

Серед можливих параметричних методів спектрального аналізу найбільшого поширення набули методи, що засновані на авторегресійній моделі формування сигналу. Це обумовлено простотою моделі, зручністю розрахунків на її основі і тим, що дана модель добре відповідає багатьом реальним завданням.

За авторегресійною моделлю сигнал $x(k)$ формується шляхом пропускання дискретного білого шуму $n(k)$ через рекурсивний фільтр N -го порядку. Спектральна щільність потужності такого сигналу має вигляд:

$$W(\omega)=\frac{\sigma_n^2}{f_\partial}\frac{1}{\left|1-a_1e^{-j\omega T}-a_2e^{-j2\omega T}-\ldots-a_Ne^{-jN\omega T}\right|^2} \quad (2.3)$$

Таким чином, даний метод зводиться до визначення коефіцієнтів моделі a_i заданого порядку N, оцінці потужності білого шуму σ_n^2 та розрахунку власне спектральної щільності потужності за формулою (2.3).

Авторегресійні методи аналізу спектру у більшості підходять для сигналів, що найбільше походять на авторегресійний процес. Гарні результати отримуються тоді, коли спектр досліджуваного процесу має чітко виражені спектральні піки. Саме тому цей варіант гарно підходить для аналізу стану голосового тракту.

Приклад оцінки спектральної щільності потужності параметричними методами наведений на рис. 2.2.3.

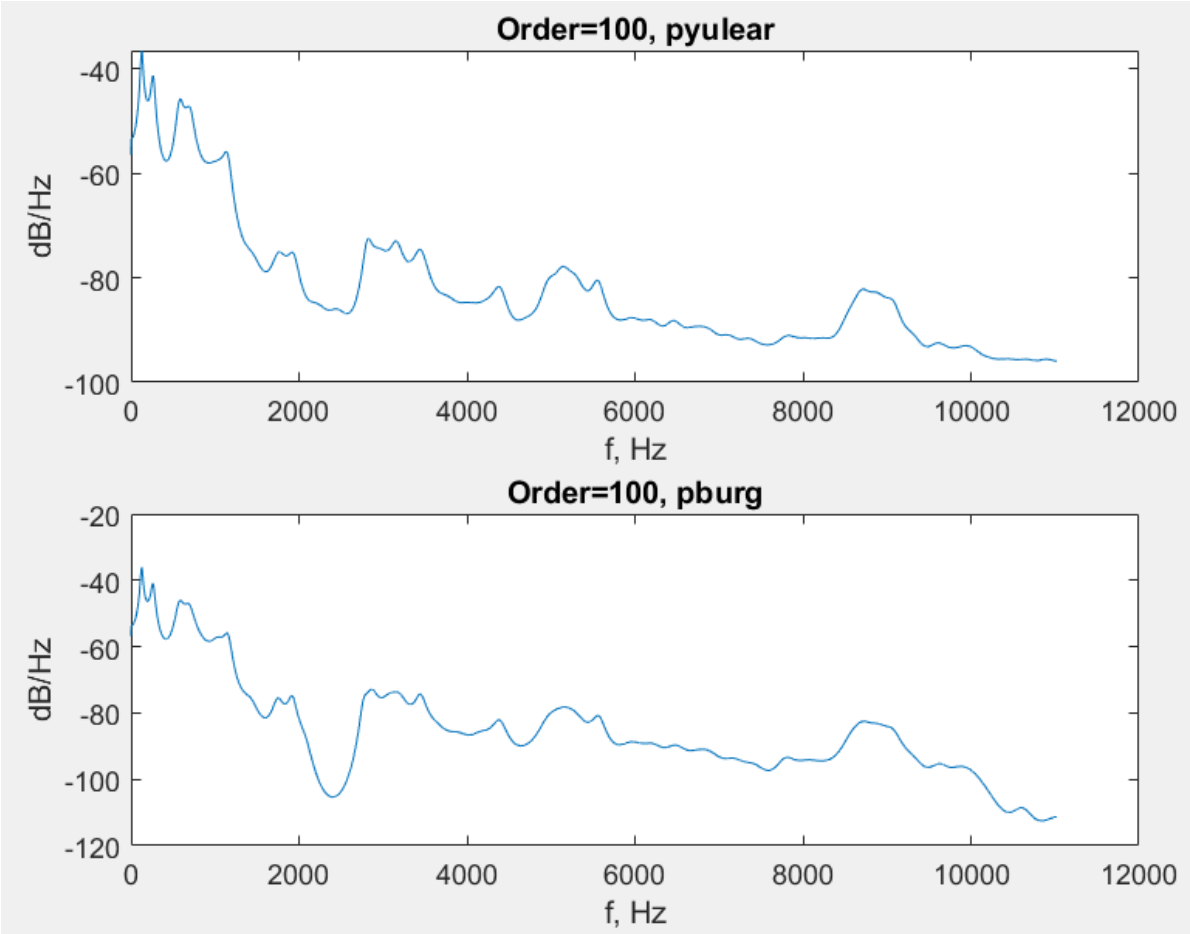

Рис. 2.2.3. Оцінка спектру методом Берга та Юла-Уолкера

Побудова:

```
a1=a(73000:73000+10000);
[Pxxx1, f1] = pyulear(a1,100,sf/2,sf);
```

```
[Pxxx2, f2] = pburg(a1,100,sf/2,sf);
figure(1);
subplot(2,1,1), plot(f1, 10*log10(Pxxx1)); title('Order=100,
pyulear'), xlabel('f, Hz'), ylabel('dB/Hz')
subplot(2,1,2), plot(f2, 10*log10(Pxxx2)); title('Order=100, pburg'),
xlabel('f, Hz'), ylabel('dB/Hz')
```

З графіків видно, що використання авторегресійної моделі в методах Берга або Юла-Уолкера дозволяє отримувати достатньо згладжений вигляд спектру при збереженні роздільної здатності.

Таким чином для реалізації поставленої задачі обрано параметричний авторегресійний метод Берга та непараметричний метод Уелча. Далі необхідно дослідити вплив параметрів оцінки та обрати оптимальні для спектрального аналізу стану голосового тракту.

Далі наводиться аналіз запису послідовності голосних фонем, вигляд якого наведено на рис. 2.2.4:

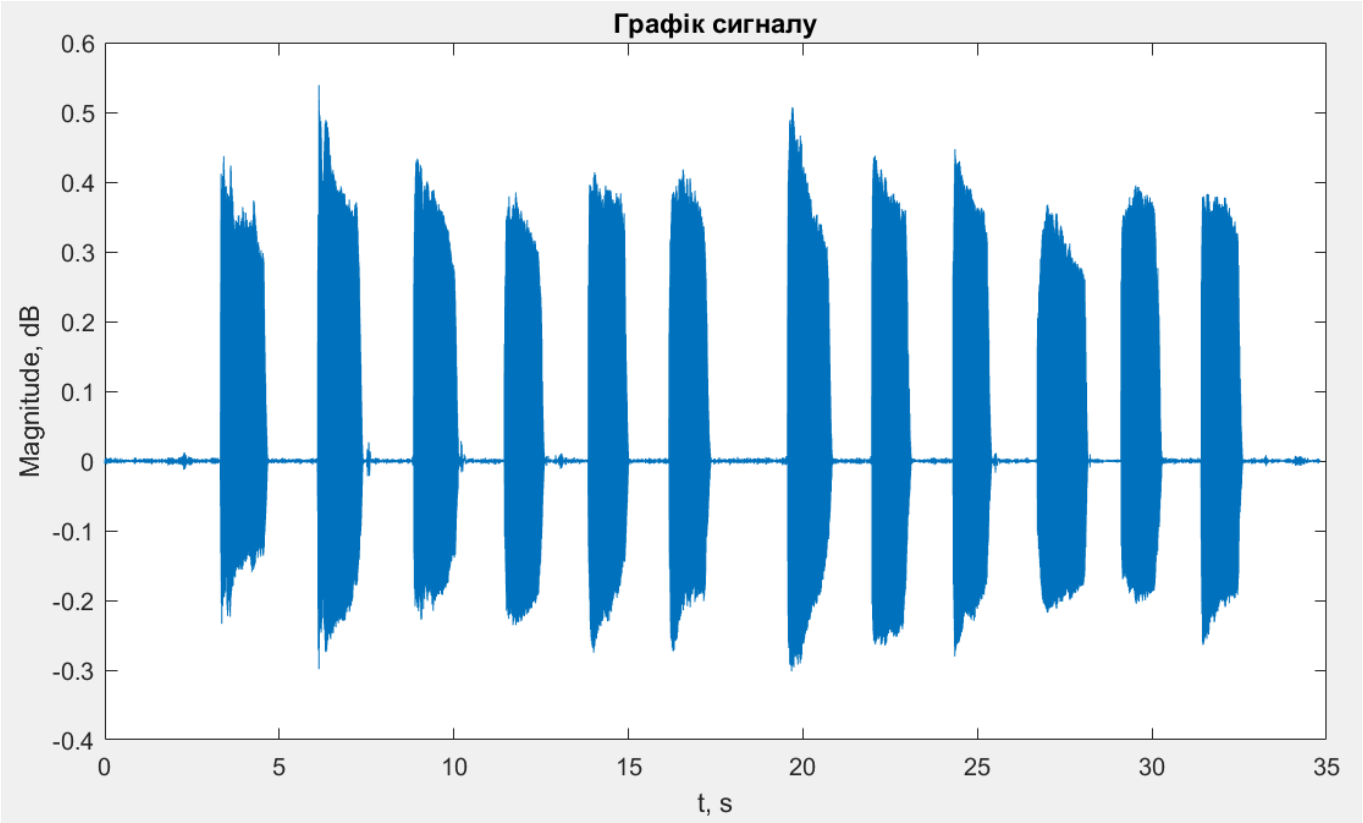

Рис. 2.2.4. Запис послідовності голосних фонем

Спершу розглянемо вплив різних вагових функцій на вигляд графіку спектру

сигналу. Для цього використовуємо апарат MATLAB у створенні вікон Бартлета-Ханна, Бартлета, Блекмена, Хеммінга, Кайзера та Гаусса-Віннера. Розмір усіх вікон приймаємо 512 відліків.

```
nfft=512;
win1 = barthannwin(nfft);
win2 = bartlett(nfft);
win3 = blackman(nfft);
win4 = hamming(nfft);
win5 = kaiser(nfft);
win6 = gausswin(nfft);
```

Далі розглянемо вплив отриманих вагових функцій на результат аналізу першої фонеми з набору методом Уелча. Обираємо для розрахунків значення параметру перекриття (‘noverlap’) половину розміру вікна:

```
[fr1,f1]=pwelch(a1,win1,nfft/2,[],sf);
[fr2,f2]=pwelch(a1,win2,nfft/2,[],sf);
[fr3,f3]=pwelch(a1,win3,nfft/2,[],sf);
[fr4,f4]=pwelch(a1,win4,nfft/2,[],sf);
[fr5,f5]=pwelch(a1,win5,nfft/2,[],sf);
[fr6,f6]=pwelch(a1,win6,nfft/2,[],sf);
```

Отримуємо графіки, зображені на рис. 2.2.5:

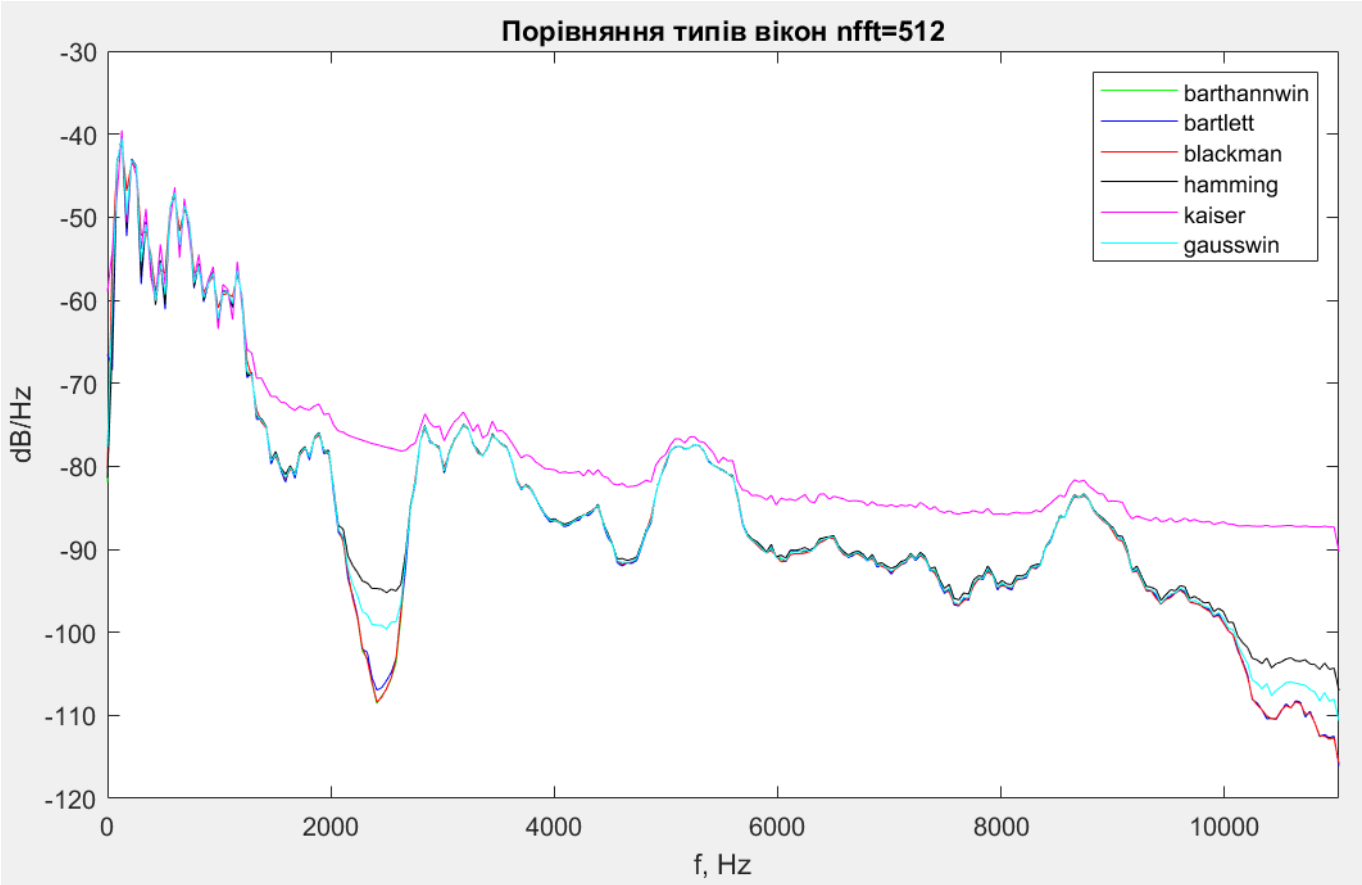

Рис. 2.2.5. Оцінка спектру голосної фонеми при розмірі вікна 512 відліків

Звідси видно, що вікно Кайзера не дає достатньої інформативності про підсилені спектральні області, що говорить про неможливість його використання. В той же час ми бачимо подібний результат для вікон Бартлета-Ханна, Бартлета, Блекмена, та окремо бачимо схожість оцінки з вікнами Хеммінга, та Гаусса-Віннера.

Щоб детальніше розглянути це питання, проведемо аналіз області спектру 20-3000 Гц. Це є ділянкою розміщення формантних областей. Сам аналіз проведемо тепер з вікнами розміром 256 відліків та для різних голосних фонем з набору. Результат зображено на рис. 2.2.6:

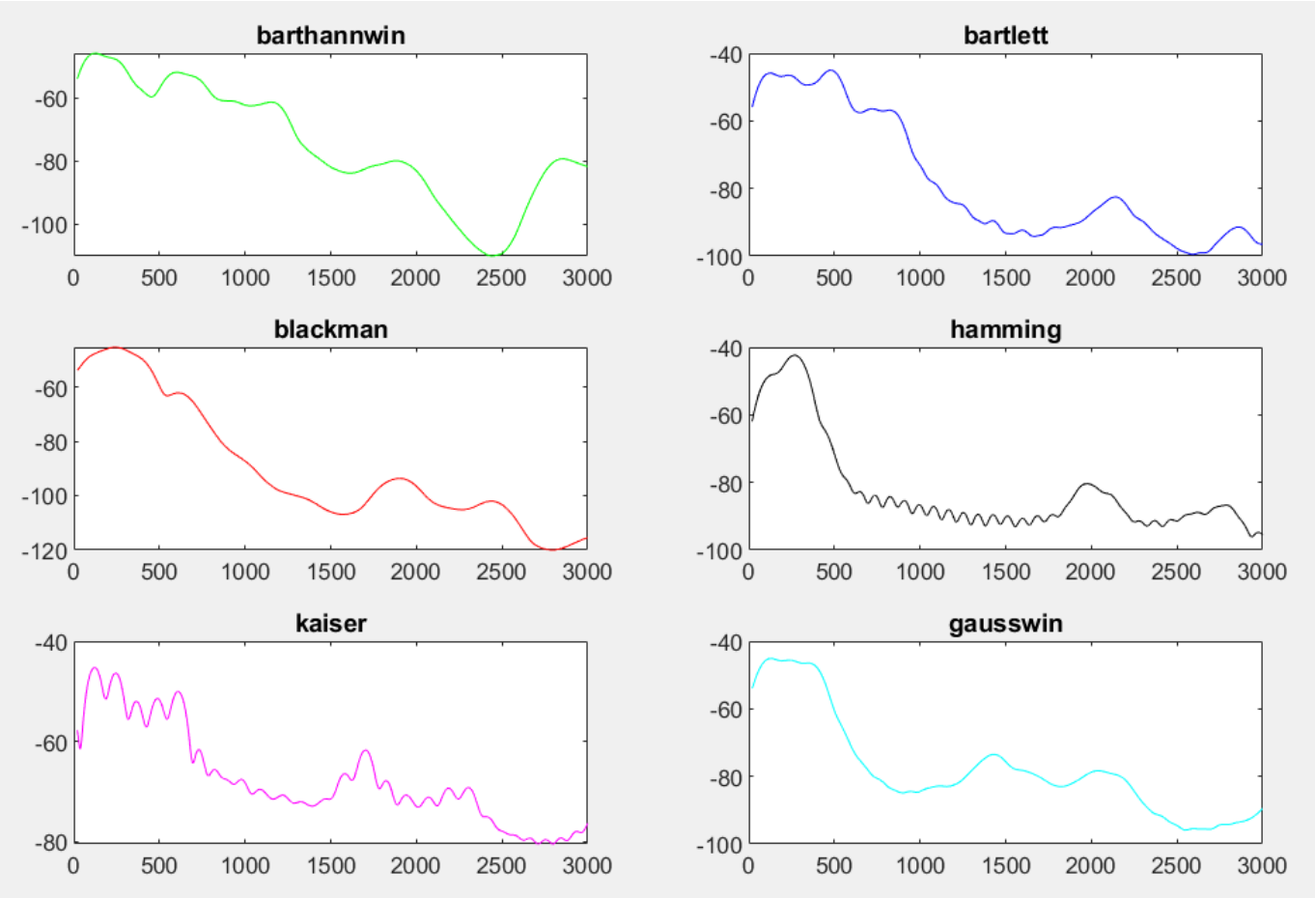

Рис. 2.2.6. Оцінка спектру різних голосних фонем при розмірі вікон 256 відліків

Можемо бачити появу флуктуацій при оцінці з використанням вікна Хеммінга та Кайзера. Гарний результат дає використання вікон Бартлета та Гаусса-Віннера.

Далі розглянемо вплив розміру вікна на результат оцінки. Для цього створюємо набори вікон Бартлета та Гаусса-Віннера розмірами 32, 64, 128, 256, 512 та 1024 відліків.

```
win11 = bartlett(1024); win12 = gausswin(1024);
```

```
win21 = bartlett(512);  win22 = gausswin(512);
win31 = bartlett(256);  win32 = gausswin(256);
win41 = bartlett(128);  win42 = gausswin(128);
win51 = bartlett(64);   win52 = gausswin(64);
win61 = bartlett(32);   win62 = gausswin(32);
```

Тоді проводимо оцінку з половинним перекриттям:

```
[fr11,f11]=pwelch(a1,win11,1024/2,[20:2:3000],sf);
[fr21,f21]=pwelch(a1,win21,512/2, [20:2:3000],sf);
[fr31,f31]=pwelch(a1,win31,256/2, [20:2:3000],sf);
[fr41,f41]=pwelch(a1,win41,128/2, [20:2:3000],sf);
[fr51,f51]=pwelch(a1,win51,64/2,  [20:2:3000],sf);
[fr61,f61]=pwelch(a1,win61,32/2,  [20:2:3000],sf);

[fr12,f12]=pwelch(a1,win12,1024/2,[20:2:3000],sf);
[fr22,f22]=pwelch(a1,win22,512/2, [20:2:3000],sf);
[fr32,f32]=pwelch(a1,win32,256/2, [20:2:3000],sf);
[fr42,f42]=pwelch(a1,win42,128/2, [20:2:3000],sf);
[fr52,f52]=pwelch(a1,win52,64/2,  [20:2:3000],sf);
[fr62,f62]=pwelch(a1,win62,32/2,  [20:2:3000],sf);
```

Результат зображено на рис. 2.2.7:

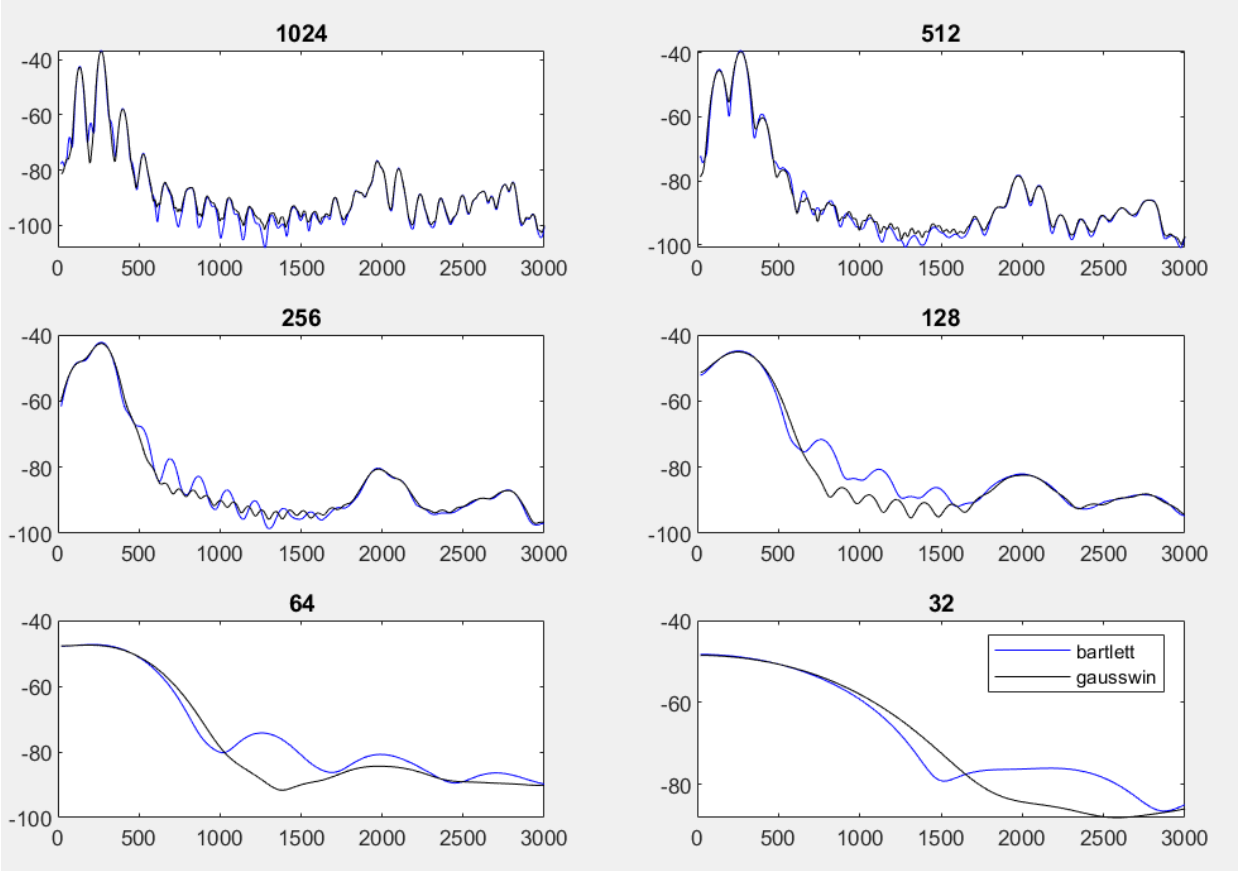

Рис. 2.2.7. Оцінка спектру голосної фонеми при змінному розмірі вікон

Звідси видно, що для користувача необхідно залишити вибір як типу вікна, так і його розміру, аби за необхідності варіювати від згладженого до більш точного вигляду оцінки спектру.

Також розглянемо параметричний авторегресивний метод Берга. В ньому змінним є параметр порядку авторегресійної моделі. Проведемо аналіз фонеми «А» з порядком моделі від 10 до 100.

```
[fr1,f1]=pburg(a1,10,20:3000,sf);
[fr2,f2]=pburg(a1,20,20:3000,sf);
[fr3,f3]=pburg(a1,40,20:3000,sf);
[fr4,f4]=pburg(a1,60,20:3000,sf);
[fr5,f5]=pburg(a1,80,20:3000,sf);
[fr6,f6]=pburg(a1,100,20:3000,sf);
```

Результат зображено на рис. 2.2.8:

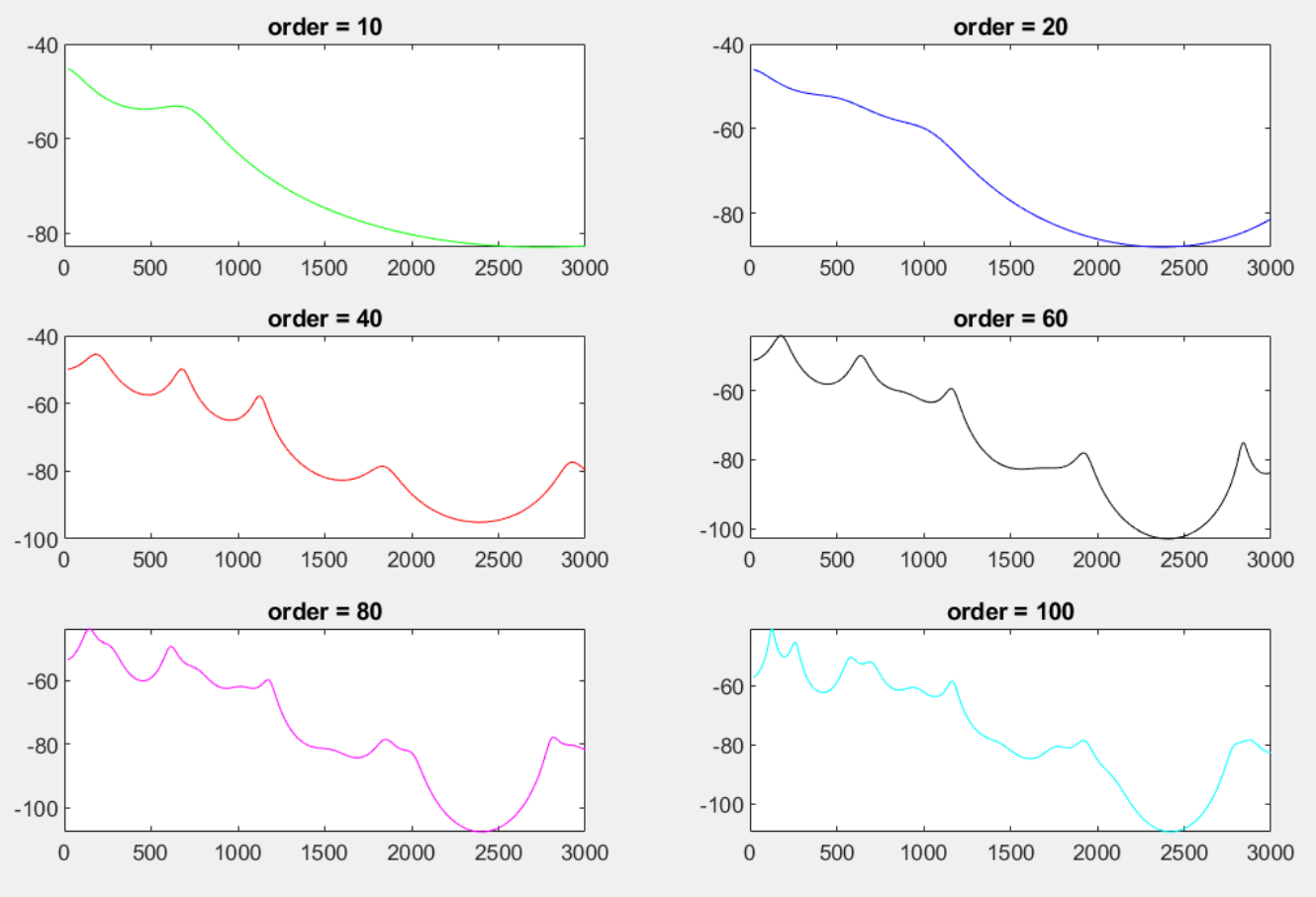

Рис. 2.2.8. Оцінка спектру голосної фонеми методом Берга при різному порядку моделі

Звідси видно, що зависокий порядок моделі призводить до розчеплення спектральних піків, в той час як малий порядок не дає достатньої інформативності про наявність підвищення потужності окремих спектральних компонент.

Проведемо повторний експеримент з різними фонемами при середньому значенні порядку моделі – 50. Результат зображено на рис. 2.2.9. З нього видно, що користувачеві необхідно залишити вибір порядку авторегресійної моделі, як і розмір вагової функції в методі Уелча.

Таким чином, нам необхідно створити m-функцію для спектрального аналізу з можливістю вибору:

1) Діапазону аналізованих частот;

2) Початку та кінця досліджуваного сигналу;

3) Типу оцінки (pburg, pwelch – gausswin, pwelch – bartlett;

4) Порядку авторегресійної моделі у випадку параметричної оцінки;

5) Розміру вікна у випадку непараметричної оцінки;

6) Вибору масштабу (логарифмічного або лінійного).

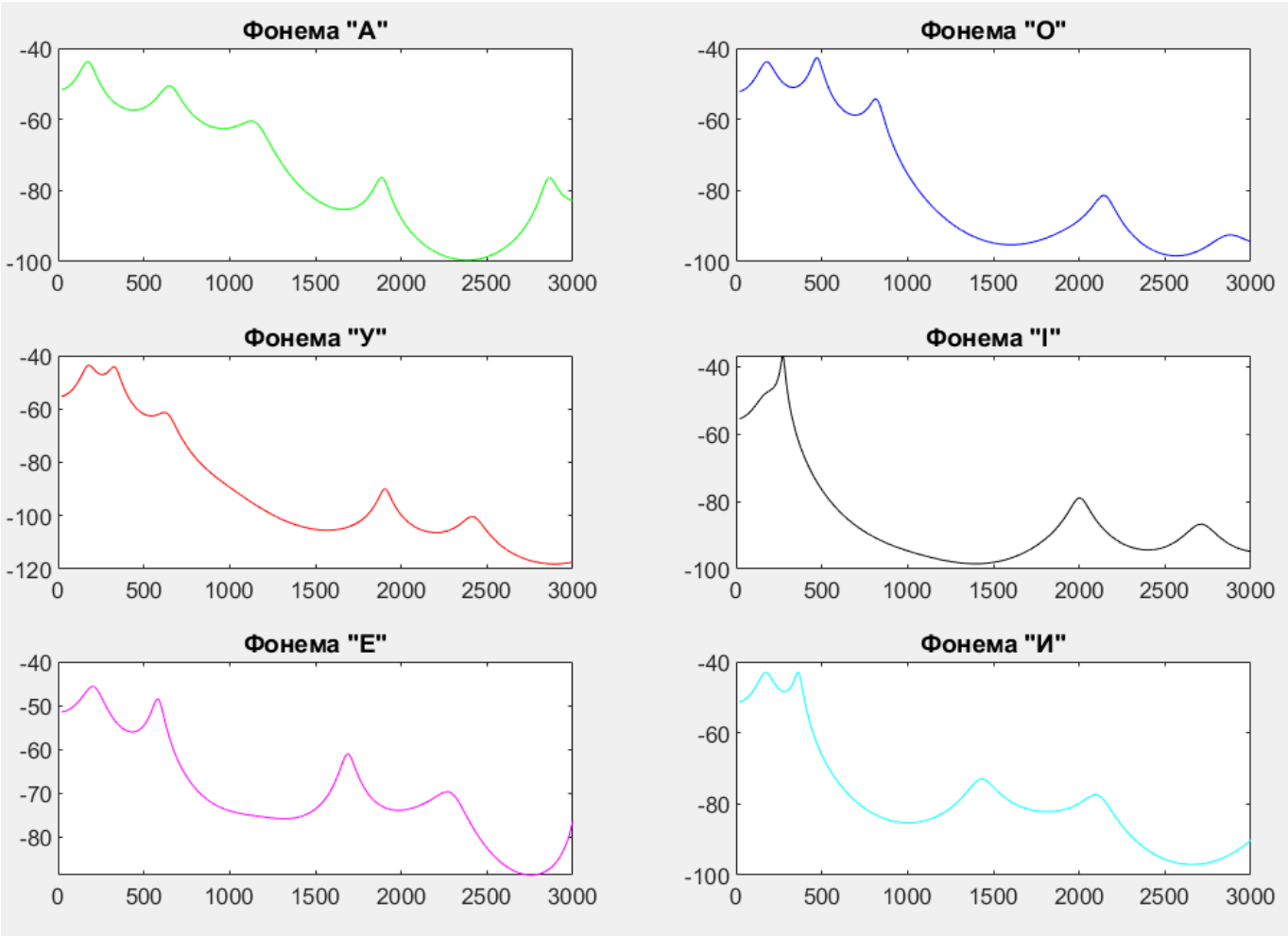

Рис. 2.2.9. Оцінка спектру голосних фонем методом Берга при порядку моделі 50

Також додаємо алгоритм видалення постійної складової та вирізання тиші з сигналів по перетину обвідної положення межі.

```
function [fr,frdB,f] = pburgfirst(SIGtall,SR,order,LeftSample,
RightSample, LeftFreq, RightFreq,type)
SIGshort=SIGtall(LeftSample:RightSample); SIGshort=SIGshort-
mean(SIGshort); SIGshort=SIGshort/max(abs(SIGshort));
[SIGenv,~]=envelope(SIGshort,100,'rms');
ii=1; SIGnew = 1;
```

```
for i=1:length(SIGshort)
    if SIGenv(i)>0.1
        SIGnew(ii)=SIGshort(i);
        ii=ii+1;
    end
end
switch type
    case 'pburg'
        [fr,f]=pburg(SIGnew,order,LeftFreq:RightFreq,SR);
        fr=fr/max(fr);
        frdB=10*log10(fr);
    case 'welch - gausswin'
        win = gausswin(order);
        [fr,f] =
pwelch(SIGnew,win,floor(order/10),[LeftFreq:RightFreq],SR);
        fr=fr/max(fr);
        frdB = 10*log10(fr);
    case 'welch - bartlett'
        win = bartlett(order);
        [fr,f] =
pwelch(SIGnew,win,floor(order/10),[LeftFreq:RightFreq],SR);
        fr=fr/max(fr);
        frdB = 10*log10(fr);
end
end
```

Для підвищення швидкодії обираємо невелике перекриття, що рівне десятій частині розміру вікна.

2.3. Методи оцінки кепстру та спектрограми

Для отримання кепстру сигналу обрано функцію rceps (дійсний кепстр) від Signal Processing Toolbox. Дійсний кепстр – це зворотнє перетворення Фур'є від натурального логарифму модулю швидкого перетворення Фур'є. Цей алгоритм реалізується наступним чином: `y = real(ifft(log(abs(fft(x)))))`.

Для дослідження кепстру важливо позбутися постійної складової вхідного сигналу та виділити лише корисний сигнал, видаливши тишу. Це робиться вже наведеним алгоритмом у функції побудови спектру.

Для виділення корисної інформації на графіку кепстру користуюсь масштабуванням:

```
[ceps,~]=rceps(SIGnew);
tvec=(1:length(ceps))/SR;
plot(tvec,ceps), xlabel('t, s'), title ('Кепстр'),
ylim([-0.1 0.5]); xlim([0 0.02]);
```

Приклад розрахунку кепстру голосної фонеми зображено на рис. 2.3.1.

Для отримання графіку спектрограми обрано функцію spectrogram. Вона заснована на використанні швидкого перетворення Фур'є. Приклад зображено на рис.2.3.2.

Рис. 2.3.1. Оцінка кепстру голосної фонеми

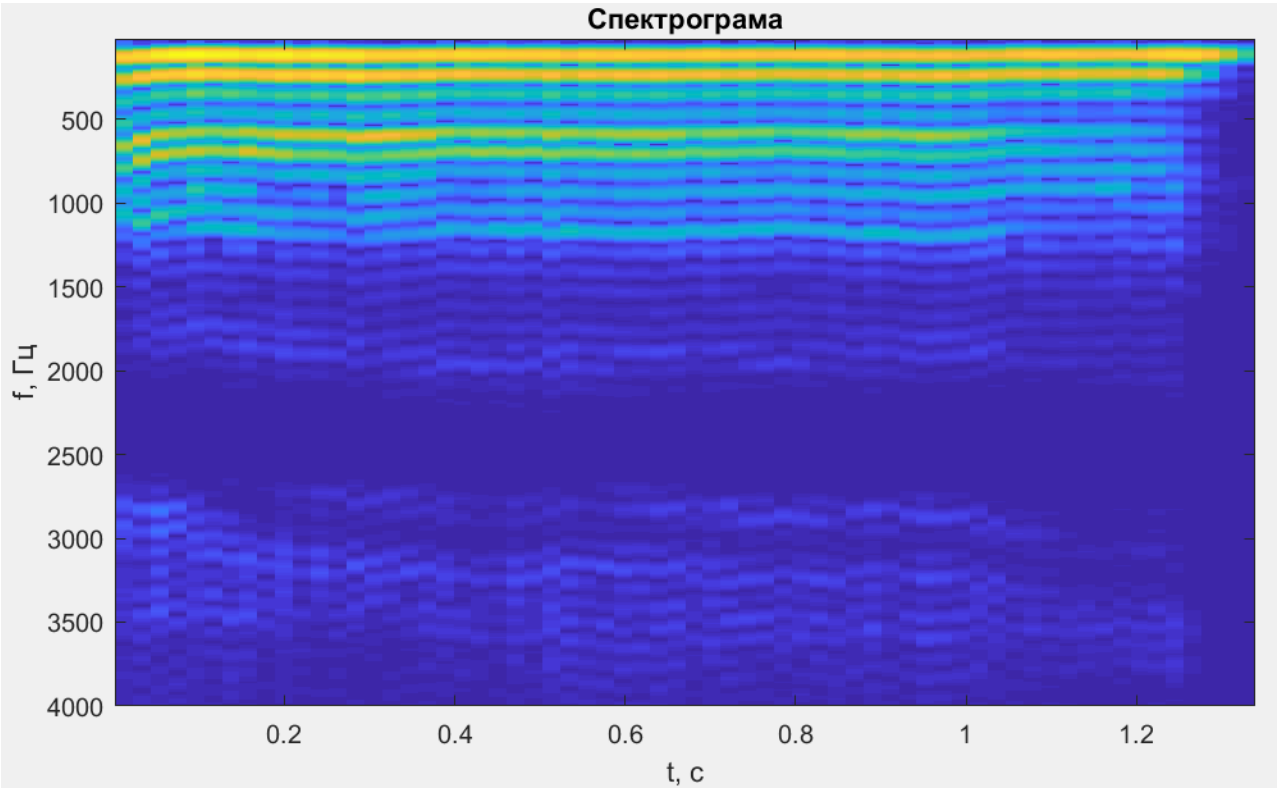

Рис. 2.3.2. Оцінка спектрограми голосної фонеми

2.4. Висновки

Програмне забезпечення MATLAB постачається з великою кількістю інструментарію, що полегшує виконання багатьох інженерних, математичних, обчислювальних питань розробки та дослідження різноманітних процесів, пов’язаних з будь-якою галуззю досліджень. На базі цього інструментарію існує велика кількість базових функцій аналізу цифрових сигналів, зокрема аудіо сигналів. Можливість створення власного інтерфейсу та компіляції отриманого коду в .EXE файл схиляє розробників до створення програмного продукту на базі MATLAB.

При розробці програмного продукту з аналізу стану голосового тракту важливо дати доступ до зміни користувачем таких параметрів, як частотного діапазону, порядку авторегресійної моделі чи розміру вагової функції – вікна, а також методу оцінки спектру.

РОЗДІЛ 3
РОЗРОБКА ПРОГРАМНОГО ІНТЕРФЕЙСУ, ОГЛЯД ОСНОВНИХ ЕТАПІВ СТВОРЕННЯ

В даному розділі наведений опис основних етапів створення користувацького програмного інтерфейсу з використанням досліджених вище можливостей та вимог.

3.1. Огляд основних елементів інтерфейсу

Кожному компоненту, що додається на робоче поле, у відповідність створюється код із заданням стандартних параметрів: положення, назви, підписів, шрифт підписів.

В основі програми лежать графічні поля, на які можна виводити побудову різних функцій, відповідно до необхідності. Ці поля створюються компонентом Axes. До його основних для нас параметрів можна віднести XLim (обмеження області визначення), YLim (обмеження області значень), YDir (напрямок зміни області визначення, що корисно для побудови спектрограми), XScale (перемикання між лінійною та логарифмічною шкалою). Якщо створити компонент Axes і надати йому назву, за його назвою у подальшому можна посилатися на будь-яке поле параметрів за необхідності. Так, наприклад, виконано зміну назви графіку відповідно до реакції компоненту на вибір.

Для подальшого опису на рис. 3.1.1. наведено вигляд отриманого інтерфейсу:

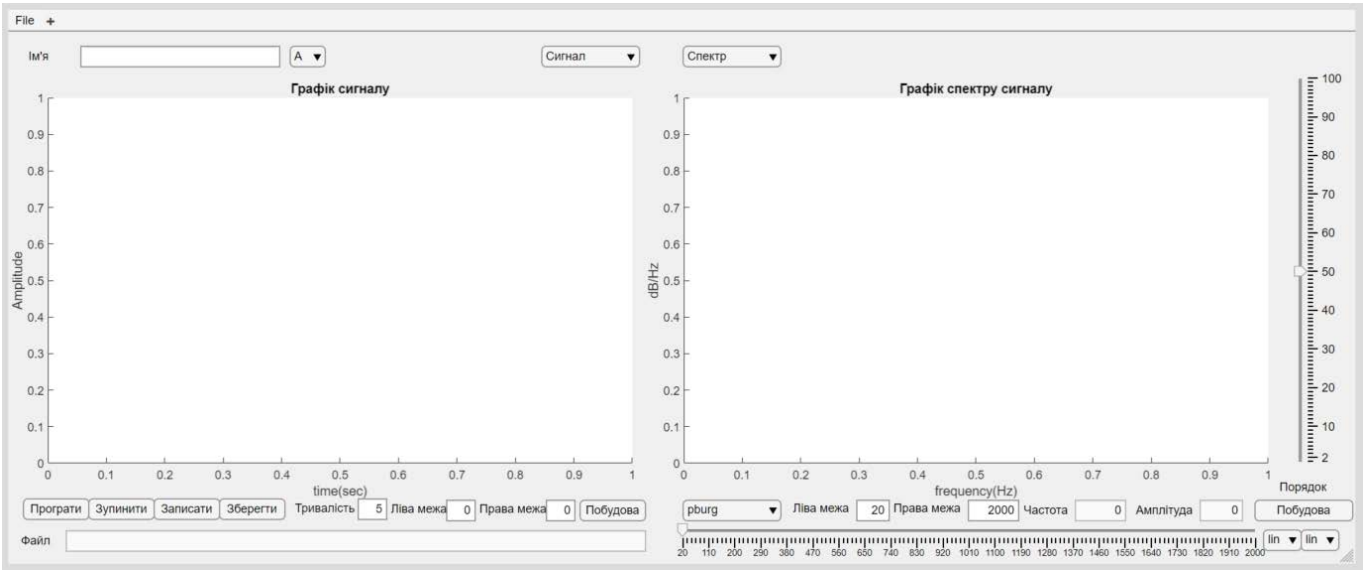

Рис. 3.1.1. Кінцевий вигляд інтерфейсу програми

Робоче поле розділене на дві частини – часову та спектральну. Це зроблено з метою контролю того, які саме дані внесено до програми. Це добре видно на графіку сигналу. Якщо запустити програму та зайти у меню файл (рис 3.1.2.) отримуємо доступ до кнопки Open.

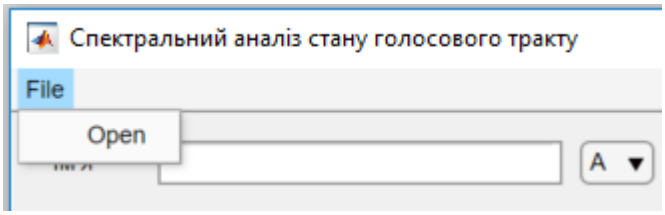

Рис. 3.1.2. Кнопка Open

Ця кнопка, як і кнопка «Записати» є основною, оскільки дає можливість внести до програми дані. При її натисканні спрацьовує Callback – функція, що виконує реакцію на дію з компонентом. Покроковий аналіз закладених дій у цю функцію наведений далі.

Спершу спрацьовує функція uigetfile – одна зі стандартних функцій реалізації діалогових вікон. Вона відповідає за виклик діалогового вікна вибору файлу, рис. 3.1.3.

```
[X,path] = uigetfile('*.wav','Оберіть WAV файл','C:\Users\Пользователь\Google Диск\Навчання\Диплом\190624_Денисенко_БакРаб\Гласные_звуки');
```

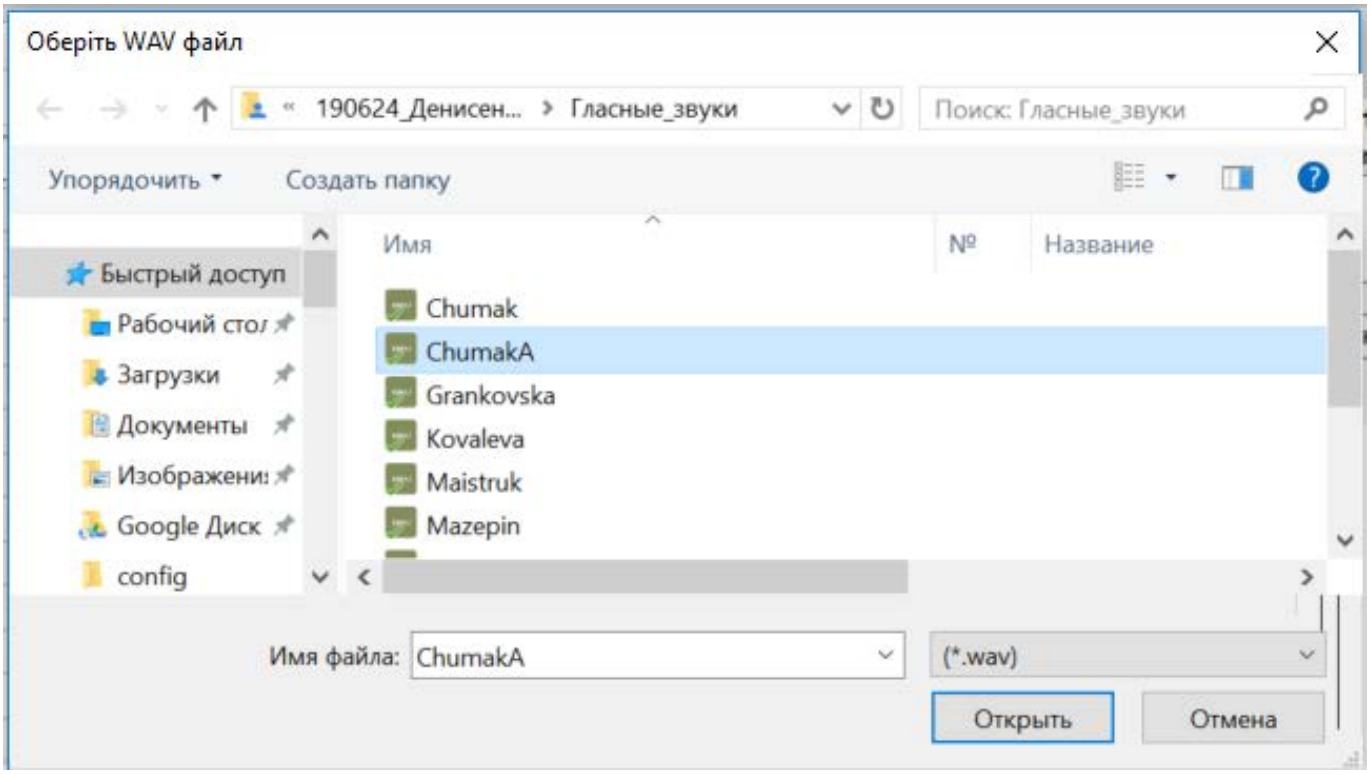

Рис. 3.1.3. Діалогове вікно вибору файлу

До параметрів функції відносяться фільтри розширень, що пропонуються для вибору, а також шлях до стандартної директорії файлів. Після натискання «Відкрити» у діалоговому вікні, до змінної X та path записується шлях до файлу та його назва. Тоді, за умови, що X не 0, тобто не натиснуто кнопки «Відміна» виконуються наступні дії.

```
if X ~= 0
```

Виконується склеювання символьних змінних X та path, а також команда audioread, що повертає від файлу за вказаною директорією масив даних та частоту дискретизації.

```
app.path2file = [path X];
[app.SIG, app.SR] = audioread(app.path2file);
```

Виконується запис у глобальні змінні початкового та кінцевого номерів відліків, а також записів до полей «Ліва межа» та «Права межа» даних про початок та кінець файлу у секундах.

```
app.LeftSample = 1;
app.EditLeftSample.Value = roundn(app.LeftSample/app.SR,-2);
app.RightSample = length(app.SIG);
app.EditRightSample.Value = roundn(app.RightSample/app.SR,-2);
```

Далі обчислюється вектор часу та будується графік сигналу на лівому полі для графіків.

```
app.tvec = (1:length(app.SIG))/app.SR;
plot(app.UIAxesTime,app.tvec(app.LeftSample:app.RightSample),app.SIG(app.LeftSample:app.
RightSample))
```

Далі встановлюється обмеження можливості вибору верхньої межі спектрального діапазону. Потім оголошуються змінні та записуються до них стандартні значення лівої та правої межі частотного діапазону.

```
app.EditRightFreq.Limits = [20 floor(app.SR/2)];
app.LeftFreq = 20;
app.EditLeftFreq.Value = app.LeftFreq;
app.RightFreq = 8000;
app.EditRightFreq.Value = app.RightFreq;
```

Наступним кроком оголошується змінна, що одночасно відповідає за порядок авторегресійної моделі в методі Берга та розмір вагової функції в методі Уелча. Також слайдер вибору порядку або розміру вікна встановлюється на це стандартне значення.

```
app.order = 50;
app.SliderOrder.Value = app.order;
```

Для полегшення роботи користувача з багатьма файлами до текстового поля записується змінна path2file, що раніше була використана для отримання даних з файлу. Таким чином на текстовому полі можна бачити повний шлях до обраного файлу.

```
app.EditFieldFile.Value = app.path2file;
```

Далі обмежується область визначення графіку по довжині файлу.

```
app.UIAxesTime.XLim = [app.LeftSample/app.SR app.RightSample/app.SR];
```

Змінна SIGis відповідає за те, що файл обраний і іншим компонентам програми є за що відповідати. Змінна cepstris вказує на те, що в даний момент для аналізу обрано спектр чи кепстр.

```
app.SIGis = 1;
app.cepstris = 0;
```

Задаються стандартні положення компонентів DropDown, що відповідають за перемикання між лінійними та логарифмічними шкалами.

```
app.YLogLin.Value = 'log';
app.XLogLin.Value = 'lin';
app.UIAxesFreq.XScale = 'linear';
```

Проводиться розрахунок спектру сигналу із стандартними параметрами та будується графік на лівому полі.

```
[fr,frdB,f] = pburgfirst(app.SIG, app.SR, app.order, app.LeftSample, app.RightSample,
app.LeftFreq, app.RightFreq, app.DropDownType.Value);
plot(app.UIAxesFreq, f,frdB)
```

Встановлюються стандартні значення крайніх частот до слайдеру, що відповідає за визначення амплітуди гармонік спектру.

```
app.SliderFreq.Limits = [app.LeftFreq app.RightFreq];
```

Налаштовується область значень та область визначення правого графіку.

```
app.UIAxesFreq.YLim = [min(frdB) max(frdB)];
app.UIAxesFreq.XLim = [app.LeftFreq app.RightFreq];
app.UIAxesFreq.YTickMode = 'auto';
```

Виконується функція побудови на графіку спектру формантних областей.

```
[~] = fonems(app.DropDownLetter.Value,fr,frdB,'log',app.UIAxesFreq);
```

Ця функція відповідає за побудову на графіку спектру прямокутників за допомогою ліній. Горизонтальні шириною в формантну область, вертикальні висотою в максимальне значення вектору fr чи frdB, в залежності від того, яка шкала обрана. Форманті області беруться за вказаними в першому розділі даними Сапожкова та Фанта. Стандартно будується для фонеми «А»

В результаті отримуємо побудову на лівому полі графіку набору голосових фонем, а на правому спектру всього сигналу рис. 3.1.4.

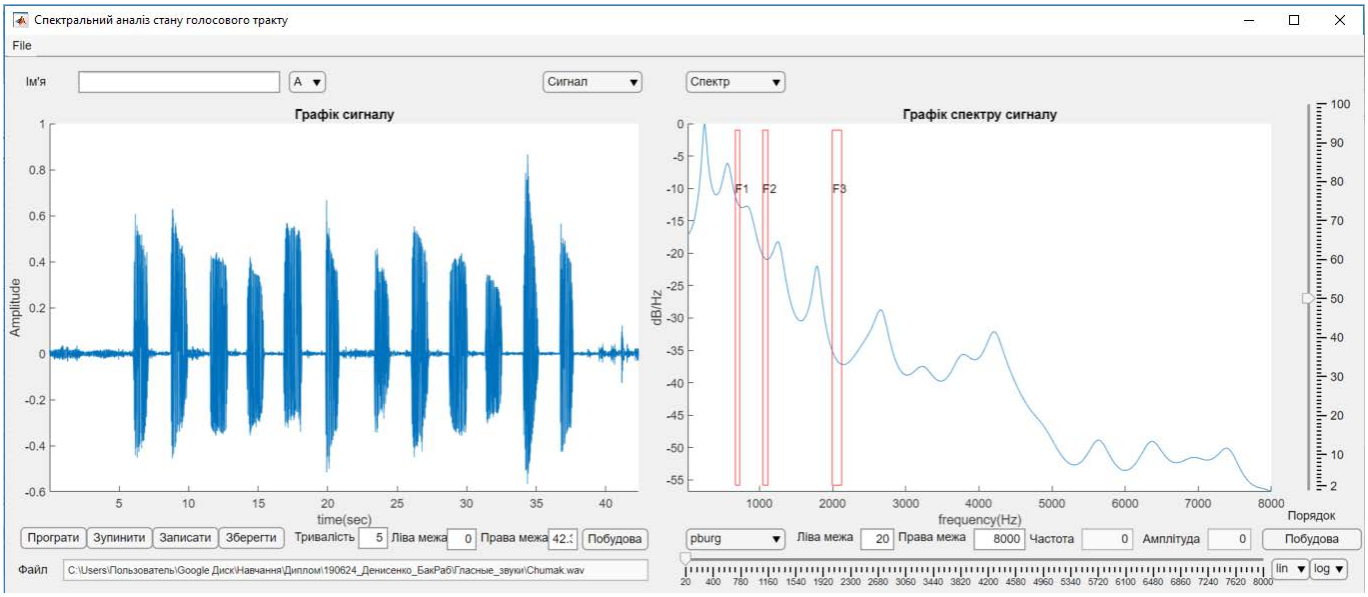

Рис. 3.1.4. Вигляд інтерфейсу після виконання кнопки Open

Далі необхідно обрати одну фонему. Для цього треба внести до полів «Ліва межа» та «Права межа» час початку та кінця необхідної фонеми з набору. Після введення даних виконуються їхні функції Callback.

```
app.LeftSample = floor(app.EditLeftSample.Value*app.SR);
```

Та окремо для правої:

```
app.RightSample = floor(app.EditRightSample.Value*app.SR);
```

За необхідності можна прослухати послідовність фонем натиснувши «Програти». Тоді виконується Callback:

```
app.aplayer = audioplayer(app.SIG(app.LeftSample:app.RightSample),app.SR,16,-1);
while (1)
play(app.aplayer);
pause((app.RightSample-app.LeftSample)/app.SR);
break
end
```

Для випадку, коли необхідно зупинити програвання запису, існує кнопка «Зупинити». Вона також відповідає за зупинку запису аудіо з мікрофону. При її натисканні виконується Callback:

```
if app.aplayer ~= 0
stop(app.aplayer);
end
if app.arecorder ~= 0
```

```
stop(app.arecorder);
end
```

Щоб внести дані до програми також існує кнопка «Записати». Її Callback подібний до кнопки «Open» з відмінністю у тому, що замість виконання діалогового вікна вибору файлу виконується функція запису аудіо з мікрофону:

```
app.SR=22050;
app.arecorder = audiorecorder (app.SR,16,1,1);
recordblocking(app.arecorder,app.EditFieldSec.Value);
app.SIG = getaudiodata(app.arecorder);
```

Далі виконуються ті самі дії запису стандартних параметрів до глобальних змінних. Це необхідно для коректної роботи програми. Потім будується спектр сигналу на правому полі. Для задання тривалості запису існує поле «Тривалість».

За необхідності можна зберегти аудіо файл. Для цього існує поле «Ім’я» куди потрібно внести дані про пацієнта та компонент DropDown для вибору фонеми (зокрема він відповідає за вибір фонеми для функції fonems). Це необхідно для формування назви файлу. Після натискання кнопки «Зберегти» виконується її Callback, де спершу записується поточний час:

```
tt = clock;
```

Потім формується змінна з назвою файлу, що складається з дати, імені та фонеми.

```
defname = [[num2str(tt(1)) '.' num2str(tt(2)) '.' num2str(tt(3)) '.' num2str(tt(4)) '.' num2str(tt(5))] '.' app.EditFieldName.Value ' ' app.DropDownLetter.Value];
```

Виконується стандартна функція виклику діалогового вікна, що повертає шлях до папки, куди збережеться файл.

```
[file, path] = uiputfile('*.wav','Оберіть шлях для збереження WAV файлу',['C:\Users\Пользователь\Google Диск\Навчання\Диплом\190624_Денисенко_БакРаб\Гласные_звуки' '\' defname]);
```

Склеюються змінні path та file, записуються дані у файл і передається повний шлях до файлу в інформаційне текстове поле.

```
app.path2file = [path file];
if file ~= 0
audiowrite(app.path2file,app.SIG(app.LeftSample:app.RightSample),app.SR);
end
app.EditFieldFile.Value = app.path2file;
```

Щоб побудувати графік сигналу у правильному масштабі після модифікування полів «Ліва межа» та «Права межа» існує ліва кнопка «Побудова». Вона необхідна для повертання графіку у правильну позицію.

```
if app.SIGis == 1
```

```
cla(app.UIAxesTime)
switch app.DropDownSig.Value
case '1'
plot(app.UIAxesTime,app.tvec(app.LeftSample:app.RightSample),app.SIG(app.LeftSample:app.
RightSample))
app.UIAxesTime.XLim = [app.LeftSample/app.SR app.RightSample/app.SR];
app.UIAxesTime.YLim = [min(app.SIG) max(app.SIG)];
case '2'
[S,F,T] = makespectrogram(app.SIG, app.SR, app.order, app.LeftSample, app.RightSample,
app.LeftFreq, app.RightFreq);
imagesc(app.UIAxesTime,T+app.LeftSample/app.SR,F,log(1+abs(S)));
app.UIAxesTime.XLim = [app.LeftSample/app.SR app.RightSample/app.SR];
app.UIAxesTime.YLim = [app.LeftFreq app.RightFreq];
set(app.UIAxesTime,'YDir', 'normal');
end
end
```

Якщо у верхньому DropDown списку обрано спектрограму, то на лівому полі будується графік спектрограми з заданими на поточний момент параметрами рис.3.1.5.

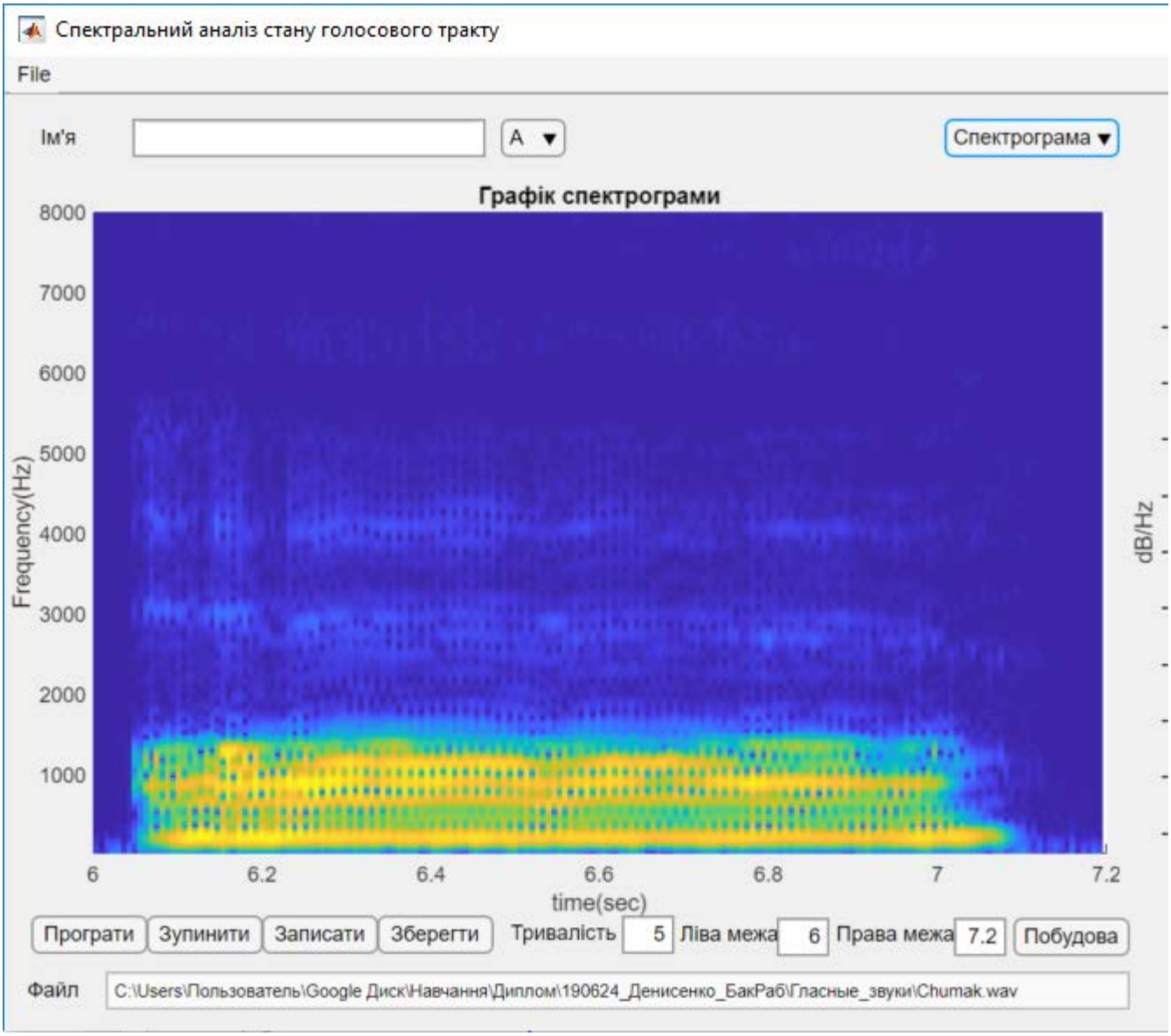

Рис. 3.1.5. Спектрограма сигналу

Також є DropDown список переходу між графіком спектру та кепстру на правому полі. Його Callback або будує заново графік спектру із заданими на поточний момент параметрами, або перемикається на кепстр, викликаючи функцію cepstr. Також важливо відмітити зміну назви графіку при виконанні функції Callback.

На спектральній частині робочого поля розміщено DropDown список, що дає вибір або параметричного методу Берга, або методу Уелча з вікнами Бартлетта або Гаусса-Віннера. У відповідь на вибір поля стирається старий графік спектру та будується новий. При цьому збивається положення слайдеру, що відповідає за порядок авторегресійної моделі або розміру вагової функції, до стандартного. А також змінюється підпис слайдера на «Порядок» або «NFFT» в залежності від обраного поля списку.

Говорячи про слайдер, після відпускання повзунку графік будується заново. Іноді це займає довгий час, при виборі великого порядку моделі, або при малих значеннях розміру вікна. Це пов’язано зі збільшенням кількості досліджуваних сегментів, що знижує швидкодію.

Нижній слайдер відповідає за вибір частоти гармоніки, амплітуда якої показується у текстовому полі «Амплітуда». Для зручного вибору частоти на графіку спектру будується вертикальна лінія на частоти поточного положення повзунка нижнього слайдеру. Для реалізації цього необхідно кожного разу стирати та будувати заново графік спектру. Алгоритм, що представлено в даній роботі не дає великої швидкодії, оскільки потребує при цьому ще й повторного розрахунку самого спектру, але при цьому є надійним. На графіку спектру це виглядає наступним чином (рис. 3.1.6):

На правій стороні робочого поля також є кнопка «Побудова», що скидає певні можливі помилки при побудові та повертає графік спектру у вірне положення.

Також додані компоненти «lin» та «log». Вони відповідають за перемикання режимів шкал: ліва за область визначення, права за область значень. Ліва просто перемикає у налаштуваннях поля Axes параметр XScale, рис. 3.1.7., а правий перемикає для інших функцій програми між вибором fr та frdB. Режим

логарифмічного масштабу по вісі X зручний для детального аналізу низьких частот спектру голосних фонем.

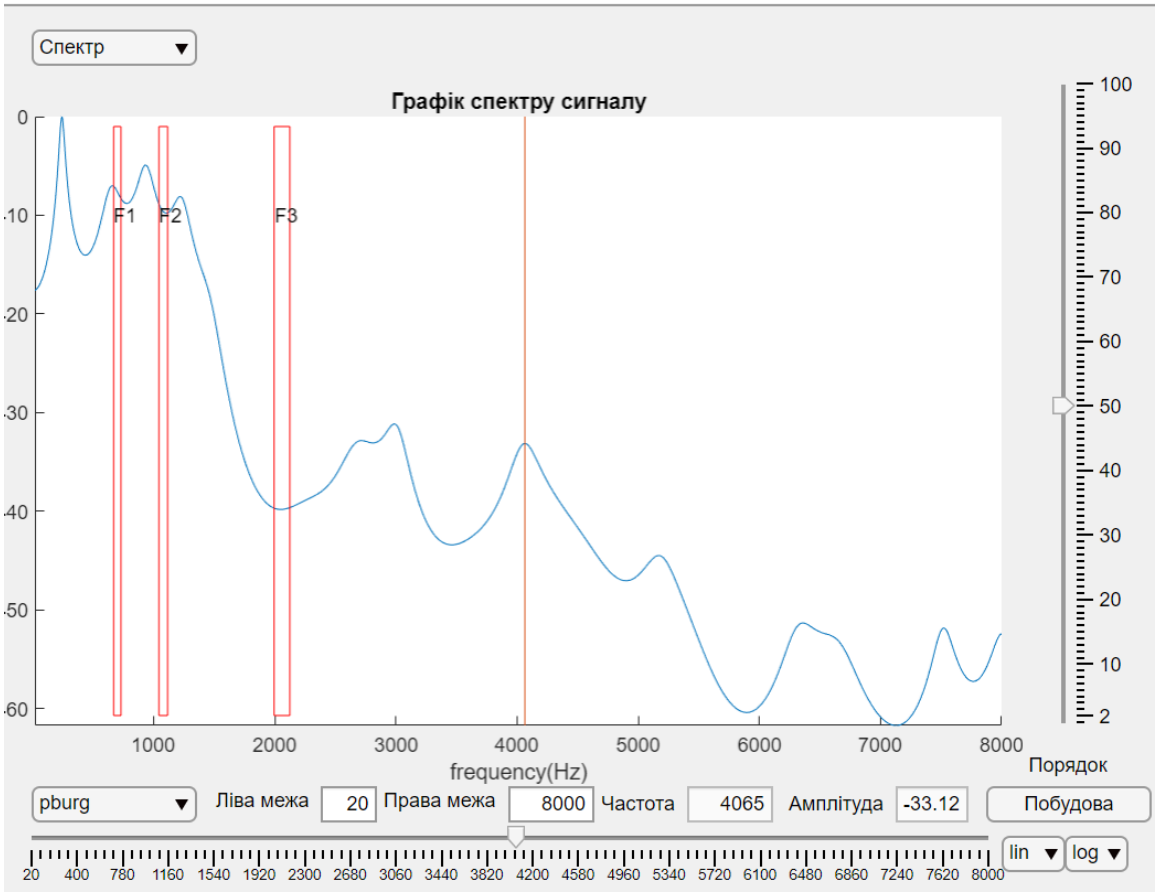

Рис. 3.1.6. Графік спектру та індикація обраної частоти

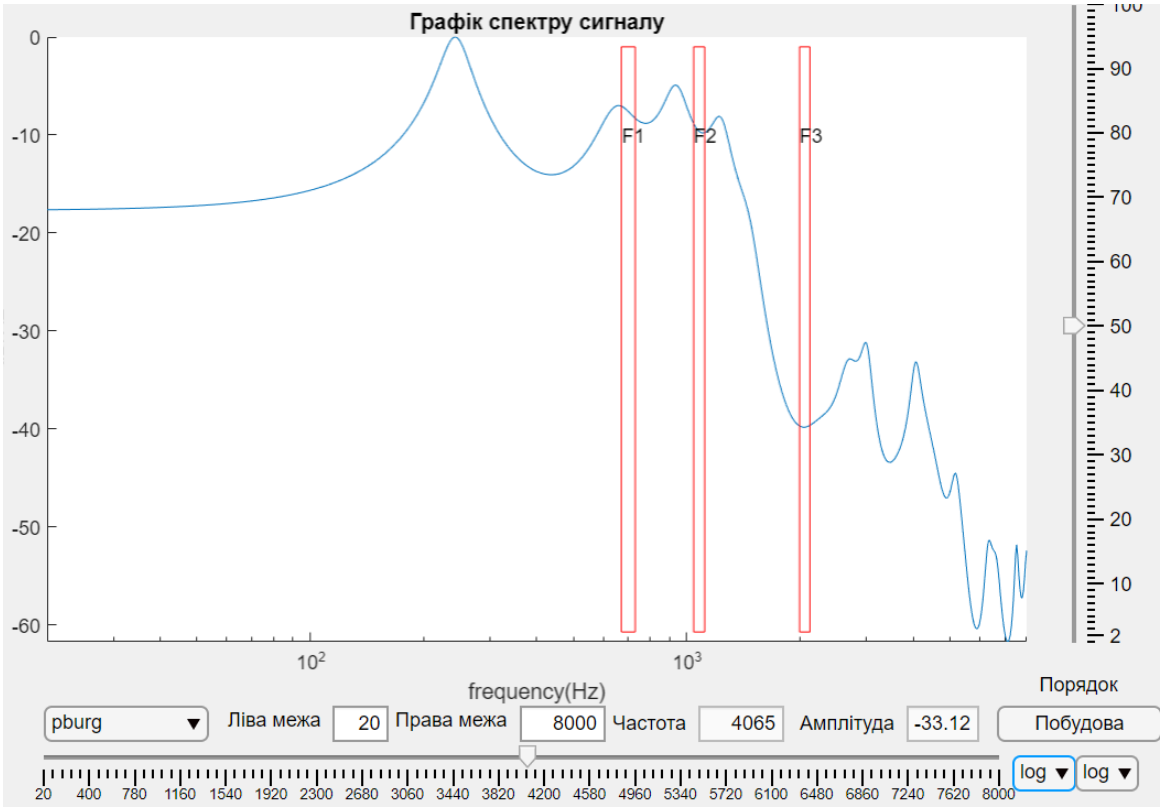

Рис. 3.1.7. Графік спектру у логарифмічному масштабі по вісі X

3.2. Висновки

Пакет AppDesigner надає великі можливості створення інтерфейсів при розробці програмного забезпечення. Маніпулюючи функціями Callback, можна доводити програму до найтонших налаштувань, що спочатку можуть здаватися непомітними, але в цілому створюють комфорт при роботі.

Важливо побудувати певний алгоритм дій кожного компоненту. Часто трапляється, що необхідно врахувати такі деталі, за які певний компонент сам по собі не відповідає. Наприклад, це може бути зміна підписів інших компонентів.

Щоб спростити написання коду, важливо створювати m-функції. Проте необхідно це враховувати при їх редагуванні після введення в основний скрипт. На жаль, іноді не все можна спростити таким чином.

ВИСНОВКИ

На даний момент поширення захворювань голосового тракту збільшилося, через зростання кількості та складності професій, що потребує його роботи. Існує велика кількість методи діагностування зв’язаних з цим патологій, але при цьому не існує дешевого, доступного та зручного для лікаря-фоніатра програмно-апаратного рішення. В силу цього на базі програмного інженерного середовища MATLAB розроблено програмну частину апаратно-програмного комплексу з аналізу стану голосового тракту.

Використовуючи програмний інструментарій, що надається MATLAB, вдалося створити зручне та гнучке в налаштуванні інтерфейсне рішення, готове для використання користувачем.

В основі його роботи лежать алгоритми ШПФ – швидкого перетворення Фур’є, що при певній модифікації поділяються на параметричні та непараметричні. В даному випадку обрано непараметричний метод Уелча та параметричний метод Берга. Користувачеві надано доступ до вибору між параметрами, необхідними для їх роботи. Для першого це розмірність вагової функції, для другого це порядок авторегресійної моделі. Все це дає широкі можливості аналізу спектру голосних фонем.

Готовий програмний продукт може бути скомпільованим та поширеним користувачам разом із стандартною безкоштовною бібліотекою MATLAB Runtime.

ПЕРЕЛІК ПОСИЛАНЬ

1. Шидловська Т. А. Функціональні порушення голосу / Т. А. Шидловська., 2011.
2. Praat: doing phonetics by computer [Електронний ресурс] – Режим доступу до ресурсу: http://www.fon.hum.uva.nl/praat/.
3. Speech Filing System [Електронний ресурс] – Режим доступу до ресурсу: https://www.phon.ucl.ac.uk/resource/sfs/.
4. Павлихин О. Г. Диагностическое значение компьютерного спектрального анализа голоса у вокалистов / О. Г. Павлихин, А. П. Мещеркин // Сборник научных трудов. Первый международный междисциплинарный конгресс "Голос" / О. Г. Павлихин, А. П. Мещеркин., 2007. – С. 86–90
5. Сапожков М.А. Речевой сигнал в кибернетике и связи. – М.: Связьиздат, 1963.
6. Покровський Н.Б. Расчет и измерение разборчивости речи. – М.: Связьиздат, 1962.
7. MathWorks - Makers of MATLAB and Simulink - MATLAB & Simulink [Електронний ресурс] – Режим доступу до ресурсу: https://www.mathworks.com/.
8. Сергиенко А. Б. Цифровая обработка сигналов / А. Б. Сергиенко., 2011. – 768 с.

Додаток А

Скрипти програмних модулів

```
classdef FirstAttempt2 < matlab.apps.AppBase
    % Properties that correspond to app components
    properties (Access = public)
        UIFigure          matlab.ui.Figure
        FileMenu          matlab.ui.container.Menu
        OpenMenu          matlab.ui.container.Menu
        LabelLeftFreq     matlab.ui.control.Label
        EditLeftFreq      matlab.ui.control.NumericEditField
        LabelRightFreq    matlab.ui.control.Label
        EditRightFreq     matlab.ui.control.NumericEditField
        ButtonBuildFreq   matlab.ui.control.Button
        LabelOrder        matlab.ui.control.Label
        SliderOrder       matlab.ui.control.Slider
        LabelLeftSample   matlab.ui.control.Label
        EditLeftSample    matlab.ui.control.NumericEditField
        LabelRightSample  matlab.ui.control.Label
        EditRightSample   matlab.ui.control.NumericEditField
        UIAxesTime        matlab.ui.control.UIAxes
        ButtonBuildTime   matlab.ui.control.Button
        SliderFreq        matlab.ui.control.Slider
        Label_6           matlab.ui.control.Label
        EditFieldAmpl     matlab.ui.control.NumericEditField
        LabelFile         matlab.ui.control.Label
        EditFieldFile     matlab.ui.control.EditField
        ButtonPlay        matlab.ui.control.Button
        ButtonStop        matlab.ui.control.Button
        ButtonRec         matlab.ui.control.Button
        DropDownType      matlab.ui.control.DropDown
        Label_8           matlab.ui.control.Label
        EditFieldSec      matlab.ui.control.NumericEditField
        Label_9           matlab.ui.control.Label
        EditFieldFreq     matlab.ui.control.NumericEditField
        ButtonSave        matlab.ui.control.Button
        Label_10          matlab.ui.control.Label
        EditFieldName     matlab.ui.control.EditField
        DropDownSpec      matlab.ui.control.DropDown
        DropDownSig       matlab.ui.control.DropDown
        XLogLin           matlab.ui.control.DropDown
        YLogLin           matlab.ui.control.DropDown
        DropDownLetter    matlab.ui.control.DropDown
        UIAxesFreq        matlab.ui.control.UIAxes
    end

    properties (Access = public)
        path2file % Description
        SIG
        SR
        LeftFreq
        RightFreq
        order
        LeftSample
        RightSample
```

```
        tvec
        SlFreq
        aplayer
        arecorder
        SIGis
        cepstris
    end

    methods (Access = private)
        % Value changed function: EditLeftFreq
        function EditLeftFreqValueChanged(app, event)
            app.LeftFreq = app.EditLeftFreq.Value;
            app.RightFreq = app.EditRightFreq.Value;
            if app.cepstris == 0
                value = app.YLogLin.Value;
                if app.SIGis == 1
                    [fr,frdB,f] =
pburgfirst(app.SIG,app.SR,floor(app.order),app.LeftSample,app.RightSample,app.LeftFreq,a
pp.RightFreq,app.DropDownType.Value);
                    switch value
                        case 'lin'
                            cla(app.UIAxesFreq);
                            plot(app.UIAxesFreq,f,fr),
                            [~] =
fonems(app.DropDownLetter.Value,fr,frdB,value,app.UIAxesFreq);
                            app.UIAxesFreq.YLim = [min(fr) max(fr)];
                        case 'log'
                            cla(app.UIAxesFreq);
                            plot(app.UIAxesFreq,f,frdB),
                            [~] =
fonems(app.DropDownLetter.Value,fr,frdB,value,app.UIAxesFreq);
                            app.UIAxesFreq.YLim = [min(frdB) max(frdB)];
                    end
                end
                app.UIAxesFreq.XLim = [app.LeftFreq app.RightFreq];
            end
            app.SliderFreq.Limits = [app.EditLeftFreq.Value app.EditRightFreq.Value];
        end
        % Value changed function: EditRightFreq
        function EditRightFreqValueChanged(app, event)
            app.LeftFreq = app.EditLeftFreq.Value;
            app.RightFreq = app.EditRightFreq.Value;
            if app.cepstris == 0
                value = app.YLogLin.Value;
                if app.SIGis == 1
                    [fr,frdB,f] =
pburgfirst(app.SIG,app.SR,floor(app.order),app.LeftSample,app.RightSample,app.LeftFreq,a
pp.RightFreq,app.DropDownType.Value);
                    switch value
                        case 'lin'
                            cla(app.UIAxesFreq);
                            plot(app.UIAxesFreq,f,fr),
                            [~] =
fonems(app.DropDownLetter.Value,fr,frdB,value,app.UIAxesFreq);
                            app.UIAxesFreq.YLim = [min(fr) max(fr)];
```

```
                        case 'log'
                            cla(app.UIAxesFreq);
                            plot(app.UIAxesFreq,f,frdB),
                            [~] =
fonems(app.DropDownLetter.Value,fr,frdB,value,app.UIAxesFreq);
                            app.UIAxesFreq.YLim = [min(frdB) max(frdB)];
                    end
                end
                app.UIAxesFreq.XLim = [app.LeftFreq app.RightFreq];
            end
            app.SliderFreq.Limits = [app.EditLeftFreq.Value app.EditRightFreq.Value];
        end
        % Menu selected function: OpenMenu
        function OpenMenuSelected(app, event)
            [X,path] = uigetfile('*.wav','Оберіть WAV
файл','C:\Users\Пользователь\Google
Диск\Навчання\Диплом\190624_Денисенко_БакРаб\Гласные_звуки');
            if X ~= 0
                app.path2file = [path X];
                [app.SIG, app.SR] = audioread(app.path2file);
                app.LeftSample = 1;                app.EditLeftSample.Value =
roundn(app.LeftSample/app.SR,-2);
                app.RightSample = length(app.SIG); app.EditRightSample.Value =
roundn(app.RightSample/app.SR,-2);
                app.tvec = (1:length(app.SIG))/app.SR;

plot(app.UIAxesTime,app.tvec(app.LeftSample:app.RightSample),app.SIG(app.LeftSample:app.
RightSample))
                app.EditRightFreq.Limits = [20 floor(app.SR/2)];
                app.LeftFreq = 20;                 app.EditLeftFreq.Value =
app.LeftFreq;
                app.RightFreq = 8000;              app.EditRightFreq.Value =
app.RightFreq;
                app.order = 50;                    app.SliderOrder.Value = app.order;
                app.EditFieldFile.Value = app.path2file;
                app.UIAxesTime.XLim = [app.LeftSample/app.SR app.RightSample/app.SR];
                app.SIGis = 1;                     app.cepstris = 0;
                app.YLogLin.Value = 'log';
                app.XLogLin.Value = 'lin';         app.UIAxesFreq.XScale = 'linear';
                [fr,frdB,f] =
pburgfirst(app.SIG,app.SR,app.order,app.LeftSample,app.RightSample,app.LeftFreq,app.Righ
tFreq,app.DropDownType.Value);
                plot(app.UIAxesFreq, f,frdB)
                app.SliderFreq.Limits = [app.LeftFreq app.RightFreq];
                app.UIAxesFreq.YLim = [min(frdB) max(frdB)];
                app.UIAxesFreq.XLim = [app.LeftFreq app.RightFreq];
                app.UIAxesFreq.YTickMode = 'auto';
                [~] = fonems(app.DropDownLetter.Value,fr,frdB,'log',app.UIAxesFreq);
            end
        end
        % Button pushed function: ButtonBuildFreq
        function ButtonBuildFreqPushed(app, event)
            if (app.SIGis == 1 && app.cepstris == 0)
```

```
                [fr,frdB,f] =
pburgfirst(app.SIG,app.SR,app.order,app.LeftSample,app.RightSample,app.LeftFreq,app.Righ
tFreq,app.DropDownType.Value);
                plot(app.UIAxesFreq, f,frdB)
                app.SliderFreq.Limits = [app.EditLeftFreq.Value
app.EditRightFreq.Value];
                switch value
                    case 'lin'
                        cla(app.UIAxesFreq);
                        plot(app.UIAxesFreq,f,fr),
                        [~] =
fonems(app.DropDownLetter.Value,fr,frdB,value,app.UIAxesFreq);
                        app.UIAxesFreq.YLim = [min(fr) max(fr)];
                    case 'log'
                        cla(app.UIAxesFreq);
                        plot(app.UIAxesFreq,f,frdB),
                        [~] =
fonems(app.DropDownLetter.Value,fr,frdB,value,app.UIAxesFreq);
                        app.UIAxesFreq.YLim = [min(frdB) max(frdB)];
                end
                app.UIAxesFreq.XLim = [app.LeftFreq app.RightFreq];
                app.UIAxesFreq.YTickMode = 'auto';


            end
        end
        % Value changed function: SliderOrder
        function SliderOrderValueChanged(app, event)
            app.order = floor(app.SliderOrder.Value);
            if (app.SIGis == 1 && app.cepstris == 0)
                [fr,frdB,f] =
pburgfirst(app.SIG,app.SR,app.order,app.LeftSample,app.RightSample,app.LeftFreq,app.Righ
tFreq,app.DropDownType.Value);
                value = app.YLogLin.Value;
                switch value
                        case 'lin'
                            cla(app.UIAxesFreq);
                            plot(app.UIAxesFreq,f,fr),
                            [~] =
fonems(app.DropDownLetter.Value,fr,frdB,value,app.UIAxesFreq);
                            app.UIAxesFreq.YLim = [min(fr) max(fr)];
                        case 'log'
                            cla(app.UIAxesFreq);
                            plot(app.UIAxesFreq,f,frdB),
                            [~] =
fonems(app.DropDownLetter.Value,fr,frdB,value,app.UIAxesFreq);
                            app.UIAxesFreq.YLim = [min(frdB) max(frdB)];
                end
                app.UIAxesFreq.XLim = [app.LeftFreq app.RightFreq];
            end
        end
        % Value changed function: EditLeftSample
        function EditLeftSampleValueChanged(app, event)
            app.LeftSample = floor(app.EditLeftSample.Value*app.SR);
        end
```

```
        % Value changed function: EditRightSample
        function EditRightSampleValueChanged(app, event)
            app.RightSample = floor(app.EditRightSample.Value*app.SR);
        end
        % Button pushed function: ButtonBuildTime
        function ButtonBuildTimePushed(app, event)
            if app.SIGis == 1
            cla(app.UIAxesTime)
            switch app.DropDownSig.Value
                case '1'

plot(app.UIAxesTime,app.tvec(app.LeftSample:app.RightSample),app.SIG(app.LeftSample:app.
RightSample))
                    app.UIAxesTime.XLim = [app.LeftSample/app.SR
app.RightSample/app.SR];
                    app.UIAxesTime.YLim = [min(app.SIG) max(app.SIG)];
                case '2'
                    [S,F,T] =
makespectrogram(app.SIG,app.SR,app.order,app.LeftSample,app.RightSample,app.LeftFreq,app
.RightFreq);
                    imagesc(app.UIAxesTime,T+app.LeftSample/app.SR,F,log(1+abs(S)));
                    app.UIAxesTime.XLim = [app.LeftSample/app.SR
app.RightSample/app.SR];
                    app.UIAxesTime.YLim = [app.LeftFreq app.RightFreq];
                    set(app.UIAxesTime,'YDir', 'normal');
            end
            end
        end
        % Value changed function: SliderFreq
        function SliderFreqValueChanged(app, event)
            if (app.SIGis == 1 && app.cepstris == 0)
                value = app.YLogLin.Value;
                app.SliderFreq.Limits = [app.EditLeftFreq.Value
app.EditRightFreq.Value];
                app.SlFreq = app.SliderFreq.Value;
                [fr,frdB,f] =
pburgfirst(app.SIG,app.SR,app.order,app.LeftSample,app.RightSample,app.LeftFreq,app.Righ
tFreq,app.DropDownType.Value);
                switch value
                        case 'lin'
                            cla(app.UIAxesFreq);
                            plot(app.UIAxesFreq,f,fr),
                            [~] =
fonems(app.DropDownLetter.Value,fr,frdB,value,app.UIAxesFreq);
                            app.UIAxesFreq.YLim = [min(fr) max(fr)];
                            hold(app.UIAxesFreq,'on');
                            plot(app.UIAxesFreq,[app.SlFreq app.SlFreq],[min(fr)
max(fr)]);
                            hold(app.UIAxesFreq,'off');
                            app.EditFieldAmpl.Value = fr(floor(app.SlFreq-
app.LeftFreq+1));
                        case 'log'
                            cla(app.UIAxesFreq);
                            plot(app.UIAxesFreq,f,frdB),
```

```
                            [~] =
fonems(app.DropDownLetter.Value,fr,frdB,value,app.UIAxesFreq);
                            app.UIAxesFreq.YLim = [min(frdB) max(frdB)];
                            hold(app.UIAxesFreq,'on');
                            plot(app.UIAxesFreq,[app.SlFreq app.SlFreq],[min(frdB)
max(frdB)]);
                            hold(app.UIAxesFreq,'off');
                            app.EditFieldAmpl.Value = frdB(floor(app.SlFreq-
app.LeftFreq+1));
                end
                app.EditFieldFreq.Value = app.SlFreq;
                app.UIAxesFreq.XLim = [app.LeftFreq app.RightFreq];
            end
        end
        % Callback function
        function ButtonPlayValueChanged(app, event)
            if isequal(1,app.ButtonPlay.value)
            a = audioplayer(app.SIG(app.LeftSample:app.RightSample),app.SR);
            play(a);
            end
        end
        % Button pushed function: ButtonPlay
        function ButtonPlayPushed(app, event)
            app.aplayer =
audioplayer(app.SIG(app.LeftSample:app.RightSample),app.SR,16,-1);
            while (1)
            play(app.aplayer);
            pause((app.RightSample-app.LeftSample)/app.SR);
            break
            end
        end
        % Button pushed function: ButtonStop
        function ButtonStopPushed(app, event)
            if app.aplayer ~= 0
                stop(app.aplayer);
            end
            if app.arecorder ~= 0
                stop(app.arecorder);
            end
        end
        % Button pushed function: ButtonRec
        function ButtonRecPushed(app, event)
            app.SR=22050;
            app.arecorder = audiorecorder (app.SR,16,1,1);
            disp('start')
            recordblocking(app.arecorder,app.EditFieldSec.Value);
            disp('stop')
            app.SIG = getaudiodata(app.arecorder);
            app.LeftSample = 1;
app.EditLeftSample.Value = roundn(app.LeftSample/app.SR,-2);
            app.RightSample = length(app.SIG);
app.EditRightSample.Value = roundn(app.RightSample/app.SR,-2);
            app.tvec = (1:length(app.SIG))/app.SR;
```

```
plot(app.UIAxesTime,app.tvec(app.LeftSample:app.RightSample),app.SIG(app.LeftSample:app.
RightSample))
            app.UIAxesTime.XLim = [app.LeftSample/app.SR app.RightSample/app.SR];
            app.UIAxesTime.YLim = [min(app.SIG) max(app.SIG)];
            app.EditRightFreq.Limits = [20 floor(app.SR/2)];
            app.LeftFreq = 20;                                      app.EditLeftFreq.Value
= app.LeftFreq;
            app.RightFreq = 8000;                                   app.EditRightFreq.Value
= app.RightFreq;
            app.order = 50;                                         app.SliderOrder.Value =
app.order;
            app.SIGis = 1;                                          app.cepstris = 0;
            app.SliderFreq.Limits = [app.LeftFreq app.RightFreq];
            [fr,frdB,f] =
pburgfirst(app.SIG,app.SR,app.order,app.LeftSample,app.RightSample,app.LeftFreq,app.Righ
tFreq,app.DropDownType.Value);
            value = app.YLogLin.Value;
            switch value
                        case 'lin'
                            cla(app.UIAxesFreq);
                            plot(app.UIAxesFreq,f,fr),
                            [~] =
fonems(app.DropDownLetter.Value,fr,frdB,value,app.UIAxesFreq);
                            app.UIAxesFreq.YLim = [min(fr) max(fr)];
                        case 'log'
                            cla(app.UIAxesFreq);
                            plot(app.UIAxesFreq,f,frdB),
                            [~] =
fonems(app.DropDownLetter.Value,fr,frdB,value,app.UIAxesFreq);
                            app.UIAxesFreq.YLim = [min(frdB) max(frdB)];
            end
            app.UIAxesFreq.XLim = [app.LeftFreq app.RightFreq];
            app.UIAxesFreq.Title.String = 'Графік спектру сигналу';
        end
        % Value changed function: DropDownType
        function DropDownTypeValueChanged(app, event)
            value = app.DropDownType.Value;
            switch value
                case {'pburg'}
                    app.LabelOrder.Text = 'Порядок';
                    app.SliderOrder.Limits = [2 100];
                    app.SliderOrder.MajorTicks = [2 10:10:100];
                    app.SliderOrder.Value = 50; app.order = 50;
                case {'welch - bartlett','welch - gauswinn'}
                    app.LabelOrder.Text = 'NFFT';
                    app.SliderOrder.Limits = [32 1024];
                    app.SliderOrder.MajorTicks = 2.^(5:10);
                    app.SliderOrder.Value = 256; app.order = 256;
            end
            if (app.SIGis == 1 && app.cepstris == 0)
                [fr,frdB,f] =
pburgfirst(app.SIG,app.SR,app.order,app.LeftSample,app.RightSample,app.LeftFreq,app.Righ
tFreq,app.DropDownType.Value);
                value1 = app.YLogLin.Value;
```

```
                switch value1
                        case 'lin'
                            cla(app.UIAxesFreq);
                            plot(app.UIAxesFreq,f,fr),
                            [~] =
fonems(app.DropDownLetter.Value,fr,frdB,value,app.UIAxesFreq);
                            app.UIAxesFreq.YLim = [min(fr) max(fr)];
                        case 'log'
                            cla(app.UIAxesFreq);
                            plot(app.UIAxesFreq,f,frdB),
                            [~] =
fonems(app.DropDownLetter.Value,fr,frdB,value,app.UIAxesFreq);
                            app.UIAxesFreq.YLim = [min(frdB) max(frdB)];
                end
                app.UIAxesFreq.XLim = [app.LeftFreq app.RightFreq];
            end
        end
        % Button pushed function: ButtonSave
        function ButtonSavePushed(app, event)
            tt = clock;
            defname = [[num2str(tt(1)) '.' num2str(tt(2)) '.' num2str(tt(3)) '.'
num2str(tt(4)) '.' num2str(tt(5))] '.' app.EditFieldName.Value ' '
app.DropDownLetter.Value];
            [file, path] = uiputfile('*.wav','Оберіть шлях для збереження WAV
файлу',['C:\Users\Пользователь\Google
Диск\Навчання\Диплом\190624_Денисенко_БакРаб\Гласные_звуки' '\' defname]);
            app.path2file = [path file];
            if file ~= 0

audiowrite(app.path2file,app.SIG(app.LeftSample:app.RightSample),app.SR);
            end
            app.EditFieldFile.Value = app.path2file;
        end
        % Value changed function: DropDownSig
        function DropDownSigValueChanged(app, event)
            if app.SIGis == 1
                cla(app.UIAxesTime)
                switch app.DropDownSig.Value
                    case '1'
                        app.tvec = (1:length(app.SIG))/app.SR;

plot(app.UIAxesTime,app.tvec(app.LeftSample:app.RightSample),app.SIG(app.LeftSample:app.
RightSample))
                        app.UIAxesTime.XLim = [app.LeftSample/app.SR
app.RightSample/app.SR];
                        app.UIAxesTime.YLim = [min(app.SIG) max(app.SIG)];
                        app.UIAxesTime.YLabel.String = 'Amplitude';
                        app.UIAxesTime.Title.String = 'Графік сигналу';
                    case '2'
                        [S,F,T] =
makespectrogram(app.SIG,app.SR,app.order,app.LeftSample,app.RightSample,app.LeftFreq,app
.RightFreq);

imagesc(app.UIAxesTime,T+app.LeftSample/app.SR,F,log10(1+abs(S)));
```

```
                        app.UIAxesTime.XLim = [app.LeftSample/app.SR
app.RightSample/app.SR];
                        app.UIAxesTime.YLim = [app.LeftFreq app.RightFreq];
                        set(app.UIAxesTime,'YDir', 'normal');
                        app.UIAxesTime.YLabel.String = 'Frequency(Hz)';
                        app.UIAxesTime.Title.String = 'Графік спектрограми';
                end
            end
        end
        % Value changed function: XLogLin
        function XLogLinValueChanged(app, event)
            if app.cepstris == 0
                value = app.XLogLin.Value;
                %[mflin,mflog] = MajorFreq (app.LeftFreq,app.RightFreq);
                switch value
                    case 'lin'
                        app.UIAxesFreq.XScale = 'linear';
                        %app.UIAxesFreq.XTickLabel = mflin;
                        %app.UIAxesFreq.XTick = mflin;
                    case 'log'
                        app.UIAxesFreq.XScale = 'log';
                        %app.UIAxesFreq.XTickLabel = mflog;
                        %app.UIAxesFreq.XTick = mflog;
                end
            end
        end
        % Value changed function: YLogLin
        function YLogLinValueChanged(app, event)
            if (app.SIGis == 1 && app.cepstris == 0)
                value = app.YLogLin.Value;
                [fr,frdB,f] =
pburgfirst(app.SIG,app.SR,app.order,app.LeftSample,app.RightSample,app.LeftFreq,app.Righ
tFreq,app.DropDownType.Value);
                switch value
                        case 'lin'
                            cla(app.UIAxesFreq);
                            plot(app.UIAxesFreq,f,fr),
                            [~] =
fonems(app.DropDownLetter.Value,fr,frdB,value,app.UIAxesFreq);
                            app.UIAxesFreq.YLim = [min(fr) max(fr)];
                        case 'log'
                            cla(app.UIAxesFreq);
                            plot(app.UIAxesFreq,f,frdB),
                            [~] =
fonems(app.DropDownLetter.Value,fr,frdB,value,app.UIAxesFreq);
                            app.UIAxesFreq.YLim = [min(frdB) max(frdB)];
                end
                app.UIAxesFreq.XLim = [app.LeftFreq app.RightFreq];
            end
        end
        % Value changed function: DropDownSpec
        function DropDownSpecValueChanged(app, event)
            if app.SIGis == 1
                value = app.DropDownSpec.Value;
                switch value
```

```
                    case '1'
                        value1 = app.YLogLin.Value;
                        [fr,frdB,f] =
pburgfirst(app.SIG,app.SR,app.order,app.LeftSample,app.RightSample,app.LeftFreq,app.Righ
tFreq,app.DropDownType.Value);
                        switch value1
                            case 'lin'
                                cla(app.UIAxesFreq);
                                plot(app.UIAxesFreq,f,fr),
                                [~] =
fonems(app.DropDownLetter.Value,fr,frdB,value,app.UIAxesFreq);
                                app.UIAxesFreq.YLim = [min(fr) max(fr)];
                            case 'log'
                                cla(app.UIAxesFreq);
                                plot(app.UIAxesFreq,f,frdB),
                                [~] =
fonems(app.DropDownLetter.Value,fr,frdB,value,app.UIAxesFreq);
                                app.UIAxesFreq.YLim = [min(frdB) max(frdB)];
                        end
                         app.UIAxesFreq.XLim = [app.LeftFreq app.RightFreq];
                         app.cepstris = 0;
                         app.UIAxesFreq.Title.String = 'Графік спектру сигналу';
                    case '2'
                        [tvecc,ceps] =
cepstr(app.SIG,app.SR,app.LeftSample,app.RightSample);
                        plot(app.UIAxesFreq,tvecc,ceps);
                        app.UIAxesFreq.YLim = [-0.1 0.5];
                        app.UIAxesFreq.XLim = [0 0.02];
                        app.cepstris = 1;
                        app.UIAxesFreq.Title.String = 'Графік кепстру сигналу';
                end
            end
        end
        % Value changed function: DropDownLetter
        function DropDownLetterValueChanged(app, event)
        if app.SIGis == 1
            value = app.YLogLin.Value;
            [fr,frdB,f] =
pburgfirst(app.SIG,app.SR,app.order,app.LeftSample,app.RightSample,app.LeftFreq,app.Righ
tFreq,app.DropDownType.Value);
            switch value
                case 'lin'
                    cla(app.UIAxesFreq);
                    plot(app.UIAxesFreq,f,fr),
                    [~] = fonems(app.DropDownLetter.Value,fr,frdB,value,app.UIAxesFreq);
                    app.UIAxesFreq.YLim = [min(fr) max(fr)];
                case 'log'
                    cla(app.UIAxesFreq);
                    plot(app.UIAxesFreq,f,frdB),
                    [~] = fonems(app.DropDownLetter.Value,fr,frdB,value,app.UIAxesFreq);
                    app.UIAxesFreq.YLim = [min(frdB) max(frdB)];
            end
        end
        app.UIAxesFreq.XLim = [app.LeftFreq app.RightFreq];
        end
```

```
    end
    % App initialization and construction
    methods (Access = private)
        % Create UIFigure and components
        function createComponents(app)
            % Create UIFigure
            app.UIFigure = uifigure;
            app.UIFigure.Position = [100 100 1360 536];
            app.UIFigure.Name = 'Спектральний аналіз стану голосового тракту';
            % Create FileMenu
            app.FileMenu = uimenu(app.UIFigure);
            app.FileMenu.Text = 'File';
            % Create OpenMenu
            app.OpenMenu = uimenu(app.FileMenu);
            app.OpenMenu.MenuSelectedFcn = createCallbackFcn(app, @OpenMenuSelected,
true);
            app.OpenMenu.Text = 'Open';
            % Create LabelLeftFreq
            app.LabelLeftFreq = uilabel(app.UIFigure);
            app.LabelLeftFreq.HorizontalAlignment = 'right';
            app.LabelLeftFreq.Position = [788 46 62 22];
            app.LabelLeftFreq.Text = 'Ліва межа';
            % Create EditLeftFreq
            app.EditLeftFreq = uieditfield(app.UIFigure, 'numeric');
            app.EditLeftFreq.ValueChangedFcn = createCallbackFcn(app,
@EditLeftFreqValueChanged, true);
            app.EditLeftFreq.Position = [857 44 34 22];
            app.EditLeftFreq.Value = 20;
            % Create LabelRightFreq
            app.LabelRightFreq = uilabel(app.UIFigure);
            app.LabelRightFreq.HorizontalAlignment = 'right';
            app.LabelRightFreq.Position = [890 46 73 22];
            app.LabelRightFreq.Text = 'Права межа';
            % Create EditRightFreq
            app.EditRightFreq = uieditfield(app.UIFigure, 'numeric');
            app.EditRightFreq.ValueChangedFcn = createCallbackFcn(app,
@EditRightFreqValueChanged, true);
            app.EditRightFreq.Position = [970 44 52 22];
            app.EditRightFreq.Value = 2000;
            % Create ButtonBuildFreq
            app.ButtonBuildFreq = uibutton(app.UIFigure, 'push');
            app.ButtonBuildFreq.ButtonPushedFcn = createCallbackFcn(app,
@ButtonBuildFreqPushed, true);
            app.ButtonBuildFreq.Position = [1259 44 100 22];
            app.ButtonBuildFreq.Text = 'Побудова';
            % Create LabelOrder
            app.LabelOrder = uilabel(app.UIFigure);
            app.LabelOrder.HorizontalAlignment = 'right';
            app.LabelOrder.Position = [1280 67 52 22];
            app.LabelOrder.Text = 'Порядок';
            % Create SliderOrder
            app.SliderOrder = uislider(app.UIFigure);
            app.SliderOrder.Limits = [1 100];
            app.SliderOrder.MajorTicks = [2 10 20 30 40 50 60 70 80 90 100];
            app.SliderOrder.Orientation = 'vertical';
```

```
            app.SliderOrder.ValueChangedFcn = createCallbackFcn(app,
@SliderOrderValueChanged, true);
            app.SliderOrder.Position = [1304 105 3 386];
            app.SliderOrder.Value = 50;
            % Create LabelLeftSample
            app.LabelLeftSample = uilabel(app.UIFigure);
            app.LabelLeftSample.HorizontalAlignment = 'right';
            app.LabelLeftSample.Position = [382 45 62 22];
            app.LabelLeftSample.Text = 'Ліва межа';
            % Create EditLeftSample
            app.EditLeftSample = uieditfield(app.UIFigure, 'numeric');
            app.EditLeftSample.ValueChangedFcn = createCallbackFcn(app,
@EditLeftSampleValueChanged, true);
            app.EditLeftSample.Position = [443 44 30 22];
            % Create LabelRightSample
            app.LabelRightSample = uilabel(app.UIFigure);
            app.LabelRightSample.HorizontalAlignment = 'right';
            app.LabelRightSample.Position = [472 45 73 22];
            app.LabelRightSample.Text = 'Права межа';
            % Create EditRightSample
            app.EditRightSample = uieditfield(app.UIFigure, 'numeric');
            app.EditRightSample.ValueChangedFcn = createCallbackFcn(app,
@EditRightSampleValueChanged, true);
            app.EditRightSample.Position = [544 44 30 22];
            % Create UIAxesTime
            app.UIAxesTime = uiaxes(app.UIFigure);
            title(app.UIAxesTime, 'Графік сигналу')
            xlabel(app.UIAxesTime, 'time(sec)')
            ylabel(app.UIAxesTime, 'Amplitude')
            app.UIAxesTime.Position = [2 67 637 424];
            % Create ButtonBuildTime
            app.ButtonBuildTime = uibutton(app.UIFigure, 'push');
            app.ButtonBuildTime.ButtonPushedFcn = createCallbackFcn(app,
@ButtonBuildTimePushed, true);
            app.ButtonBuildTime.Position = [578 44 67 22];
            app.ButtonBuildTime.Text = 'Побудова';
            % Create SliderFreq
            app.SliderFreq = uislider(app.UIFigure);
            app.SliderFreq.Limits = [20 2000];
            app.SliderFreq.ValueChangedFcn = createCallbackFcn(app,
@SliderFreqValueChanged, true);
            app.SliderFreq.FontSize = 9;
            app.SliderFreq.Position = [682 33 578 3];
            app.SliderFreq.Value = 20;
            % Create Label_6
            app.Label_6 = uilabel(app.UIFigure);
            app.Label_6.HorizontalAlignment = 'right';
            app.Label_6.Position = [1129 44 68 22];
            app.Label_6.Text = 'Амплітуда';
            % Create EditFieldAmpl
            app.EditFieldAmpl = uieditfield(app.UIFigure, 'numeric');
            app.EditFieldAmpl.Editable = 'off';
            app.EditFieldAmpl.Position = [1204 44 44 22];
            % Create LabelFile
            app.LabelFile = uilabel(app.UIFigure);
```

```
            app.LabelFile.HorizontalAlignment = 'right';
            app.LabelFile.Position = [9 13 35 22];
            app.LabelFile.Text = 'Файл';
            % Create EditFieldFile
            app.EditFieldFile = uieditfield(app.UIFigure, 'text');
            app.EditFieldFile.Editable = 'off';
            app.EditFieldFile.FontSize = 10;
            app.EditFieldFile.Position = [59 13 586 22];
            % Create ButtonPlay
            app.ButtonPlay = uibutton(app.UIFigure, 'push');
            app.ButtonPlay.ButtonPushedFcn = createCallbackFcn(app, @ButtonPlayPushed,
true);
            app.ButtonPlay.Position = [16 45 67 22];
            app.ButtonPlay.Text = 'Програти';
            % Create ButtonStop
            app.ButtonStop = uibutton(app.UIFigure, 'push');
            app.ButtonStop.ButtonPushedFcn = createCallbackFcn(app, @ButtonStopPushed,
true);
            app.ButtonStop.Position = [82 45 67 22];
            app.ButtonStop.Text = 'Зупинити';
            % Create ButtonRec
            app.ButtonRec = uibutton(app.UIFigure, 'push');
            app.ButtonRec.ButtonPushedFcn = createCallbackFcn(app, @ButtonRecPushed,
true);
            app.ButtonRec.Position = [148 45 67 22];
            app.ButtonRec.Text = 'Записати';
            % Create DropDownType
            app.DropDownType = uidropdown(app.UIFigure);
            app.DropDownType.Items = {'pburg', 'welch - bartlett', 'welch - gausswin'};
            app.DropDownType.ValueChangedFcn = createCallbackFcn(app,
@DropDownTypeValueChanged, true);
            app.DropDownType.Position = [682 44 100 22];
            app.DropDownType.Value = 'pburg';
            % Create Label_8
            app.Label_8 = uilabel(app.UIFigure);
            app.Label_8.HorizontalAlignment = 'right';
            app.Label_8.Position = [285 46 66 22];
            app.Label_8.Text = 'Тривалість';
            % Create EditFieldSec
            app.EditFieldSec = uieditfield(app.UIFigure, 'numeric');
            app.EditFieldSec.Position = [354 45 30 22];
            app.EditFieldSec.Value = 5;
            % Create Label_9
            app.Label_9 = uilabel(app.UIFigure);
            app.Label_9.HorizontalAlignment = 'right';
            app.Label_9.Position = [1021 44 50 22];
            app.Label_9.Text = 'Частота';
            % Create EditFieldFreq
            app.EditFieldFreq = uieditfield(app.UIFigure, 'numeric');
            app.EditFieldFreq.Editable = 'off';
            app.EditFieldFreq.Position = [1078 44 52 22];
            % Create ButtonSave
            app.ButtonSave = uibutton(app.UIFigure, 'push');
            app.ButtonSave.ButtonPushedFcn = createCallbackFcn(app, @ButtonSavePushed,
true);
```

```
            app.ButtonSave.Position = [214 45 67 22];
            app.ButtonSave.Text = 'Зберегти';
            % Create Label_10
            app.Label_10 = uilabel(app.UIFigure);
            app.Label_10.HorizontalAlignment = 'right';
            app.Label_10.Position = [16 501 26 22];
            app.Label_10.Text = 'Ім''я';
            % Create EditFieldName
            app.EditFieldName = uieditfield(app.UIFigure, 'text');
            app.EditFieldName.Position = [74 501 202 22];
            % Create DropDownSpec
            app.DropDownSpec = uidropdown(app.UIFigure);
            app.DropDownSpec.Items = {'Спектр', 'Кепстр'};
            app.DropDownSpec.ItemsData = {'1', '2'};
            app.DropDownSpec.ValueChangedFcn = createCallbackFcn(app,
@DropDownSpecValueChanged, true);
            app.DropDownSpec.Position = [682 501 100 22];
            app.DropDownSpec.Value = '1';
            % Create DropDownSig
            app.DropDownSig = uidropdown(app.UIFigure);
            app.DropDownSig.Items = {'Сигнал', 'Спектрограма'};
            app.DropDownSig.ItemsData = {'1', '2'};
            app.DropDownSig.ValueChangedFcn = createCallbackFcn(app,
@DropDownSigValueChanged, true);
            app.DropDownSig.Position = [539 501 100 22];
            app.DropDownSig.Value = '1';
            % Create XLogLin
            app.XLogLin = uidropdown(app.UIFigure);
            app.XLogLin.Items = {'lin', 'log'};
            app.XLogLin.ValueChangedFcn = createCallbackFcn(app, @XLogLinValueChanged,
true);
            app.XLogLin.Position = [1268 14 39 22];
            app.XLogLin.Value = 'lin';
            % Create YLogLin
            app.YLogLin = uidropdown(app.UIFigure);
            app.YLogLin.Items = {'lin', 'log'};
            app.YLogLin.ValueChangedFcn = createCallbackFcn(app, @YLogLinValueChanged,
true);
            app.YLogLin.Position = [1306 14 39 22];
            app.YLogLin.Value = 'lin';
            % Create DropDownLetter
            app.DropDownLetter = uidropdown(app.UIFigure);
            app.DropDownLetter.Items = {'А', 'Е', 'И', 'І', 'О', 'У'};
            app.DropDownLetter.ValueChangedFcn = createCallbackFcn(app,
@DropDownLetterValueChanged, true);
            app.DropDownLetter.Position = [285 501 37 22];
            app.DropDownLetter.Value = 'А';
            % Create UIAxesFreq
            app.UIAxesFreq = uiaxes(app.UIFigure);
            title(app.UIAxesFreq, 'Графік спектру сигналу')
            xlabel(app.UIAxesFreq, 'frequency(Hz)')
            ylabel(app.UIAxesFreq, 'dB/Hz')
            app.UIAxesFreq.Position = [644 67 637 424];
        end
    end
```

```
    methods (Access = public)
        % Construct app
        function app = FirstAttempt2
            % Create and configure components
            createComponents(app)
            % Register the app with App Designer
            registerApp(app, app.UIFigure)
            if nargout == 0
                clear app
            end
        end
        % Code that executes before app deletion
        function delete(app)
            % Delete UIFigure when app is deleted
            delete(app.UIFigure)
        end
    end
end


function [fr,frdB,f] =
pburgfirst(SIGtall,SR,order,LeftSample,RightSample,LeftFreq,RightFreq,type)
SIGshort=SIGtall(LeftSample:RightSample); SIGshort=SIGshort-mean(SIGshort);
SIGshort=SIGshort/max(abs(SIGshort));
[SIGenv,~]=envelope(SIGshort,100,'rms');
ii=1; SIGnew = 1;
for i=1:length(SIGshort)
    if SIGenv(i)>0.1
        SIGnew(ii)=SIGshort(i);
        ii=ii+1;
    end
end
switch type
    case 'pburg'
        [fr,f]=pburg(SIGnew,order,LeftFreq:RightFreq,SR);
        fr=fr/max(fr);
        frdB=10*log10(fr);
    case 'welch - gausswin'
        win = gausswin(order);
        [fr,f] = pwelch(SIGnew,win,floor(order/10),[LeftFreq:RightFreq],SR);
        fr=fr/max(fr);
        frdB = 10*log10(fr);
    case 'welch - bartlett'
        win = bartlett(order);
        [fr,f] = pwelch(SIGnew,win,floor(order/10),[LeftFreq:RightFreq],SR);
        fr=fr/max(fr);
        frdB = 10*log10(fr);
end
end


function [tvec,ceps] = cepstr(SIGtall,SR,LeftSample,RightSample)
SIGshort=SIGtall(LeftSample:RightSample); SIGshort=SIGshort-mean(SIGshort);
SIGshort=SIGshort/max(abs(SIGshort));
[SIGenv,~]=envelope(SIGshort,100,'rms');
ii=1;
for i=1:length(SIGshort)
    if SIGenv(i)>0.1
        SIGnew(ii)=SIGshort(i);
```

```
        ii=ii+1;
    end
end
[ceps,~]=rceps(SIGnew);
tvec=(1:length(ceps))/SR;
end


function [s,f,t] =
makespectrogram(SIG,SR,order,LeftSample,RightSample,LeftFreq,RightFreq)
win1 = bartlett(order);
[s,f,t] =
spectrogram(SIG(LeftSample:RightSample),win1,floor(order/10),LeftFreq:RightFreq,
SR,'yaxis');
end


function [a] = fonems(letter,fr,frdB,type,UIAxesFreq)
switch letter
    case 'A'
        f1 = 700-30; f11 = 700+30;   f2 = 1080-36;   f22 = 1080+36;   f3 = 2060-
65;    f33 = 2060+65;
    case 'E'
        f1 = 440-20;  f11 = 440+20;    f2 = 1800-48;  f22 = 1800+48;  f3 = 2550-
65;    f33 = 2550+65;
    case 'И'
        f1 = 300-22;  f11 = 300+22;    f2 = 1480-60;  f22 = 1480+60;  f3 = 2330-
40;    f33 = 2330+40;
    case 'I'
        f1 = 240-30;  f11 = 240+30;    f2 = 2250-38;  f22 = 2250+38;  f3 = 3200-
120;   f33 = 3200+120;
    case 'O'
        f1 = 535-27;  f11 = 535+27;    f2 = 780-32;   f22 = 780+32;   f3 = 2500-
50;    f33 = 2500+50;
    case 'У'
        f1 = 300-35;  f11 = 300+35;    f2 = 625-25;   f22 = 625+25;   f3 = 2500-
55;    f33 = 2500+55;
end
hold(UIAxesFreq,'on');
switch type
    case 'lin'
        plot(UIAxesFreq, [f1 f11],[max(fr)-0.05 max(fr)-0.05],'r'),
plot(UIAxesFreq, [f1 f1],[max(fr)-0.05 min(fr)+0.05],'r'), plot(UIAxesFreq, [f11
f11],[max(fr)-0.05 min(fr)+0.05],'r'), plot(UIAxesFreq, [f1 f11],[min(fr)+0.05
min(fr)+0.05],'r'),
        text(UIAxesFreq, f1+1, max(fr)*0.8, 'F1');
        plot(UIAxesFreq, [f2 f22],[max(fr)-0.05 max(fr)-0.05],'r'),
plot(UIAxesFreq, [f2 f2],[max(fr)-0.05 min(fr)+0.05],'r'), plot(UIAxesFreq, [f22
f22],[max(fr)-0.05 min(fr)+0.05],'r'), plot(UIAxesFreq, [f2 f22],[min(fr)+0.05
min(fr)+0.05],'r'),
        text(UIAxesFreq, f2+1, max(fr)*0.8, 'F2');
        plot(UIAxesFreq, [f3 f33],[max(fr)-0.05 max(fr)-0.05],'r'),
plot(UIAxesFreq, [f3 f3],[max(fr)-0.05 min(fr)+0.05],'r'), plot(UIAxesFreq, [f33
f33],[max(fr)-0.05 min(fr)+0.05],'r'), plot(UIAxesFreq, [f3 f33],[min(fr)+0.05
min(fr)+0.05],'r'),
        text(UIAxesFreq, f3+1, max(fr)*0.8, 'F3');
    case 'log'
        plot(UIAxesFreq, [f1 f11],[max(frdB)-1 max(frdB)-1],'r'),
plot(UIAxesFreq, [f1 f1],[max(frdB)-1 min(frdB)+1],'r'), plot(UIAxesFreq, [f11
```

```
f11],[max(frdB)-1 min(frdB)+1],'r'), plot(UIAxesFreq, [f1 f11],[min(frdB)+1
min(frdB)+1],'r'),
        text(UIAxesFreq, f1+1, max(frdB)-10, 'F1');
        plot(UIAxesFreq, [f2 f22],[max(frdB)-1 max(frdB)-1],'r'),
plot(UIAxesFreq, [f2 f2],[max(frdB)-1 min(frdB)+1],'r'), plot(UIAxesFreq, [f22
f22],[max(frdB)-1 min(frdB)+1],'r'), plot(UIAxesFreq, [f2 f22],[min(frdB)+1
min(frdB)+1],'r'),
        text(UIAxesFreq, f2+1, max(frdB)-10, 'F2');
        plot(UIAxesFreq, [f3 f33],[max(frdB)-1 max(frdB)-1],'r'),
plot(UIAxesFreq, [f3 f3],[max(frdB)-1 min(frdB)+1],'r'), plot(UIAxesFreq, [f33
f33],[max(frdB)-1 min(frdB)+1],'r'), plot(UIAxesFreq, [f3 f33],[min(frdB)+1
min(frdB)+1],'r'),
        text(UIAxesFreq, f3+1, max(frdB)-10, 'F3');
end
hold(UIAxesFreq,'off');
a=1;
end
```